SEMINAR NO. FH3000

NOS INTERACTIVE TERMINAL USAGE
VERSION 2

Student Handout

PROPRIETARY NOTICE

The ideas and designs set forth in this document are the property of Control Data Corporation and are not to be disseminated, distributed, or otherwise conveyed to third persons without the express written permission of Control Data Corporation.

REVISION RECORD	
REVISION	**DESCRIPTION**
A	Manual Release
B	Manual Update
(04-15-80)	
C	Manual Update
(07-15-82)	
D	Manual Update to Version 2
(06-01-84)	
Publication No. FH3000-1	

REVISION LETTERS I, O, Q AND X ARE NOT USED

NOS INTERACTIVE TERMINAL USAGE

Address comments concerning this manual to:

CONTROL DATA CORPORATION
National Coordinator
Bloomington Facility (MNA02B)
5001 West 80th Street
Bloomington, Minnesota 55437

or use Comment Sheet in the back of this manual.

©COPYRIGHT CONTROL DATA CORPORATION
All Rights Reserved

GENERAL COURSE DESCRIPTION

COURSE TITLE: NOS V2 INTERACTIVE TERMINAL USAGE

COURSE NUMBER: FH3000-1

COURSE LENGTH: 3 days

DESCRIPTION:

This course introduces the student to the operating system, file concepts and commands and job processing commands.

PREREQUISITES:

Knowledge of FORTRAN or BASIC

COURSE OBJECTIVES:

Upon completion, the student will be expected to demonstrate a knowledge of program entry and execution, command processing, terminal control, timesharing and permanent file commands, program and text editing and batch usage from an interactive terminal.

LESSON 1

INTRODUCTION

LESSON PREVIEW:

This lesson introduces the student to the CYBER hardware, operating system, and terminal support and characteristics.

OBJECTIVES:

After completing this lesson the student should be able to:

- Name the components of a CYBER
- Explain the control point concept
- Identify parts of the terminal
- Introduce terminal control

TRAINING AIDS:

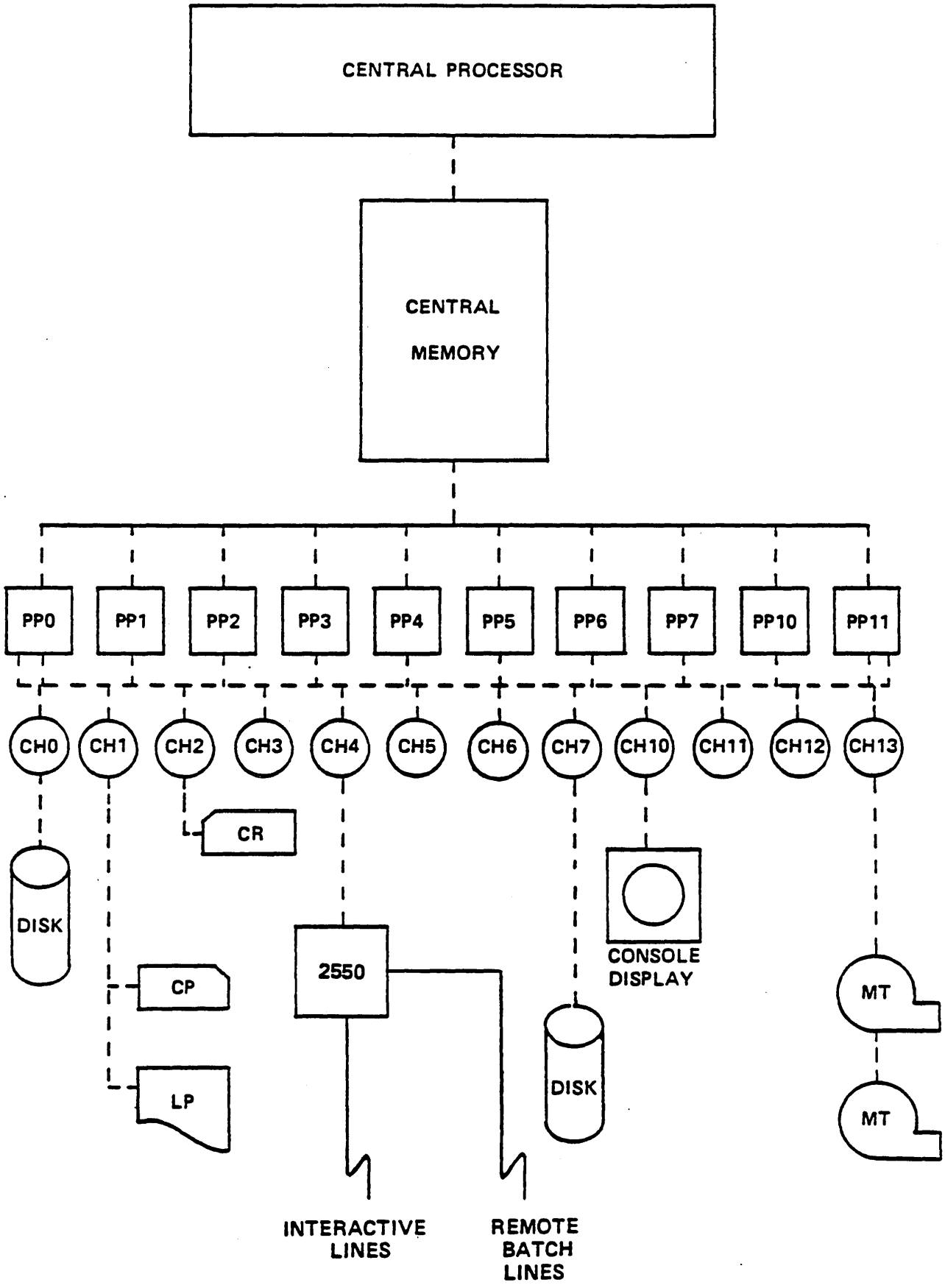
Visuals

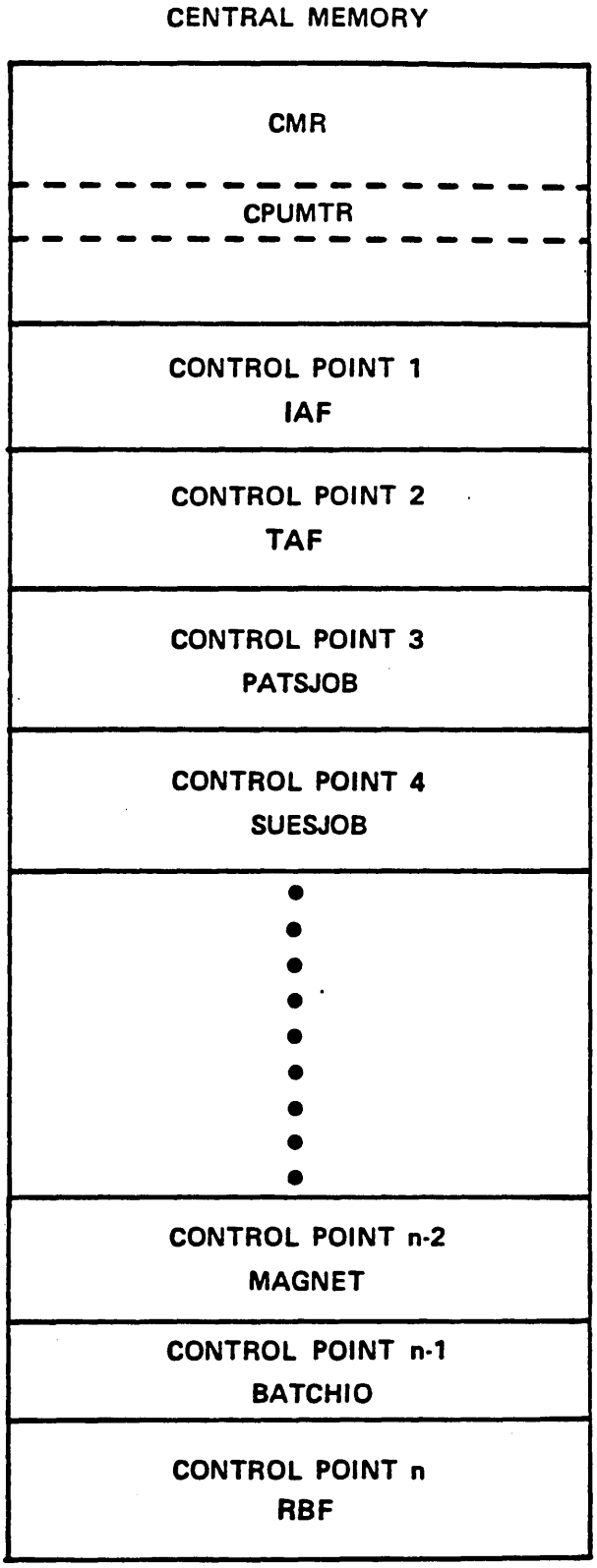
PROJECTS:

None

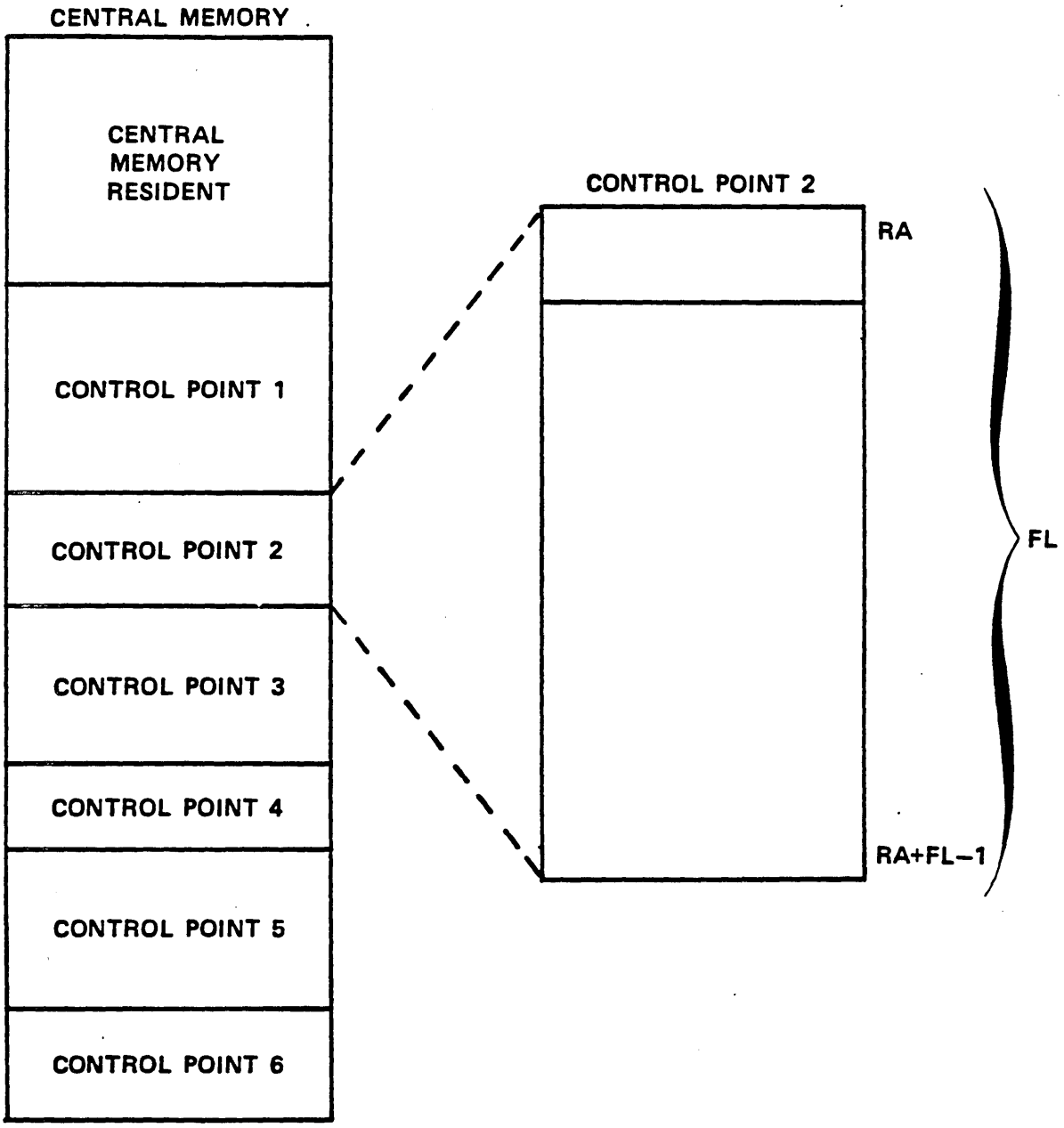
REFERENCES:

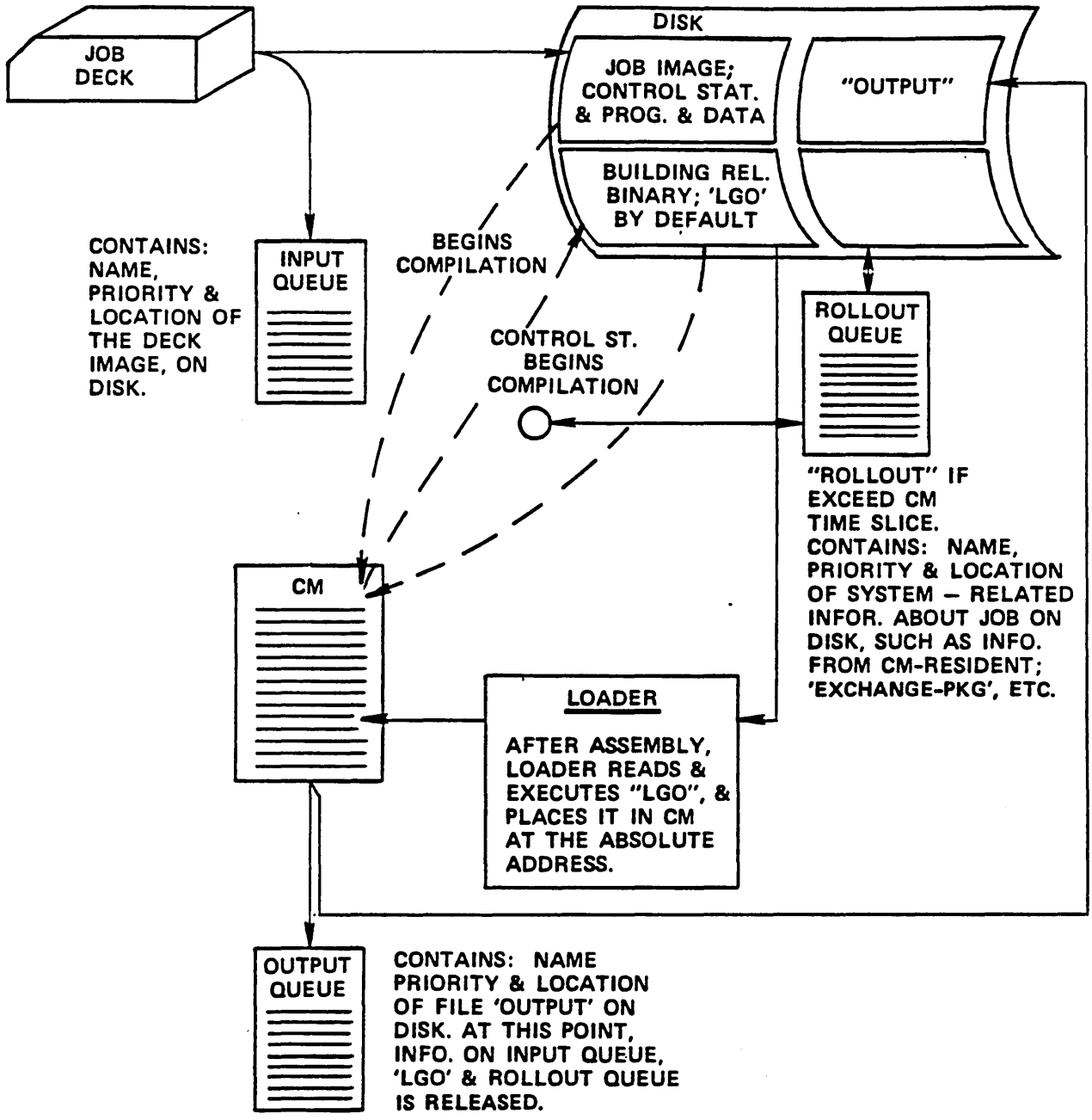
Interactive Facility Reference Manual Chapters 1, 2, Appendix F


CENTRAL PROCESSOR
CENTRAL
MEMORY
PP0
PP1
PP2
PP3
PP4
PP5
PP6
PP7
PP10
PP11
CH0
CH1
CH2
CH3
CH4
CH5
CH6
CH7
CH10
CH11
CH12
CH13
CR
DISK
CP
LP
2550
CONSOLE
DISPLAY
DISK
MT
MT
INTERACTIVE
LINES
REMOTE
BATCH
LINES


CENTRAL MEMORY
CMR
CPUMTR
CONTROL POINT 1
IAF
CONTROL POINT 2
TAF
CONTROL POINT 3
PATSJOB
CONTROL POINT 4
SUESJOB
CONTROL POINT n-2
MAGNET
CONTROL POINT n-1
BATCHIO
CONTROL POINT n
RBF

CONTROL POINT CONCEPT




JOB DECK
DISK
JOB IMAGE; CONTROL STAT. & PROG. & DATA
"OUTPUT"
BUILDING REL. BINARY; 'LGO' BY DEFAULT
CONTAINS: NAME, PRIORITY & LOCATION OF THE DECK IMAGE, ON DISK.
INPUT QUEUE
BEGINS COMPILATION
CONTROL ST. BEGINS COMPILATION
ROLLOUT QUEUE
"ROLLOUT" IF EXCEED CM TIME SLICE. CONTAINS: NAME, PRIORITY & LOCATION OF SYSTEM – RELATED INFOR. ABOUT JOB ON DISK, SUCH AS INFO. FROM CM-RESIDENT; 'EXCHANGE-PKG', ETC.
CM
LOADER
AFTER ASSEMBLY, LOADER READS & EXECUTES "LGO", & PLACES IT IN CM AT THE ABSOLUTE ADDRESS.
OUTPUT QUEUE
CONTAINS: NAME PRIORITY & LOCATION OF FILE 'OUTPUT' ON DISK. AT THIS POINT, INFO. ON INPUT QUEUE, 'LGO' & ROLLOUT QUEUE IS RELEASED.

NOS TIMESHARING ATTRIBUTES

- Excellent response time for the interactive user with simultaneous processing of local and remote batch jobs.
- Large number of terminals supported.
- Line editor available which provides file editing capabilities for the terminal user.
- Interactive compiler languages.
- Ability to execute previously compiled programs stored in the permanent file subsystem.
- Control language and procedure files which provide greater flexibility in the processing of a series of control cards.

MODES OF COMPUTER OPERATION

- BATCH

  A. All information computer needs to process entire task is given at one time (i.e., a job)

     1. Operating system control
     2. Program control
     3. Data

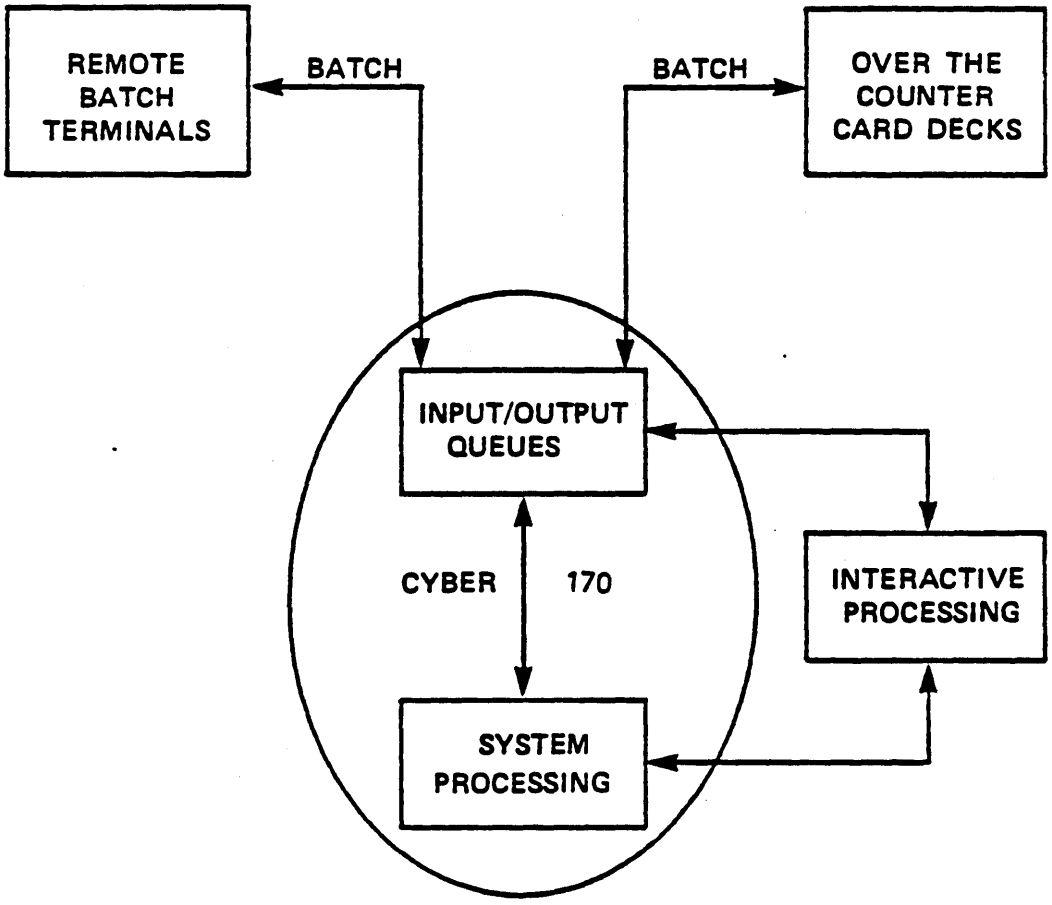
  B. All results sent back to user at one time.

- INTERACTIVE

  A. User gives information to the computer one line at a time.

  B. Computer replies sent back in response to each line of information. (A "Conversation" takes place between user and the computer. User supplied information and computer responses are interspersed.)

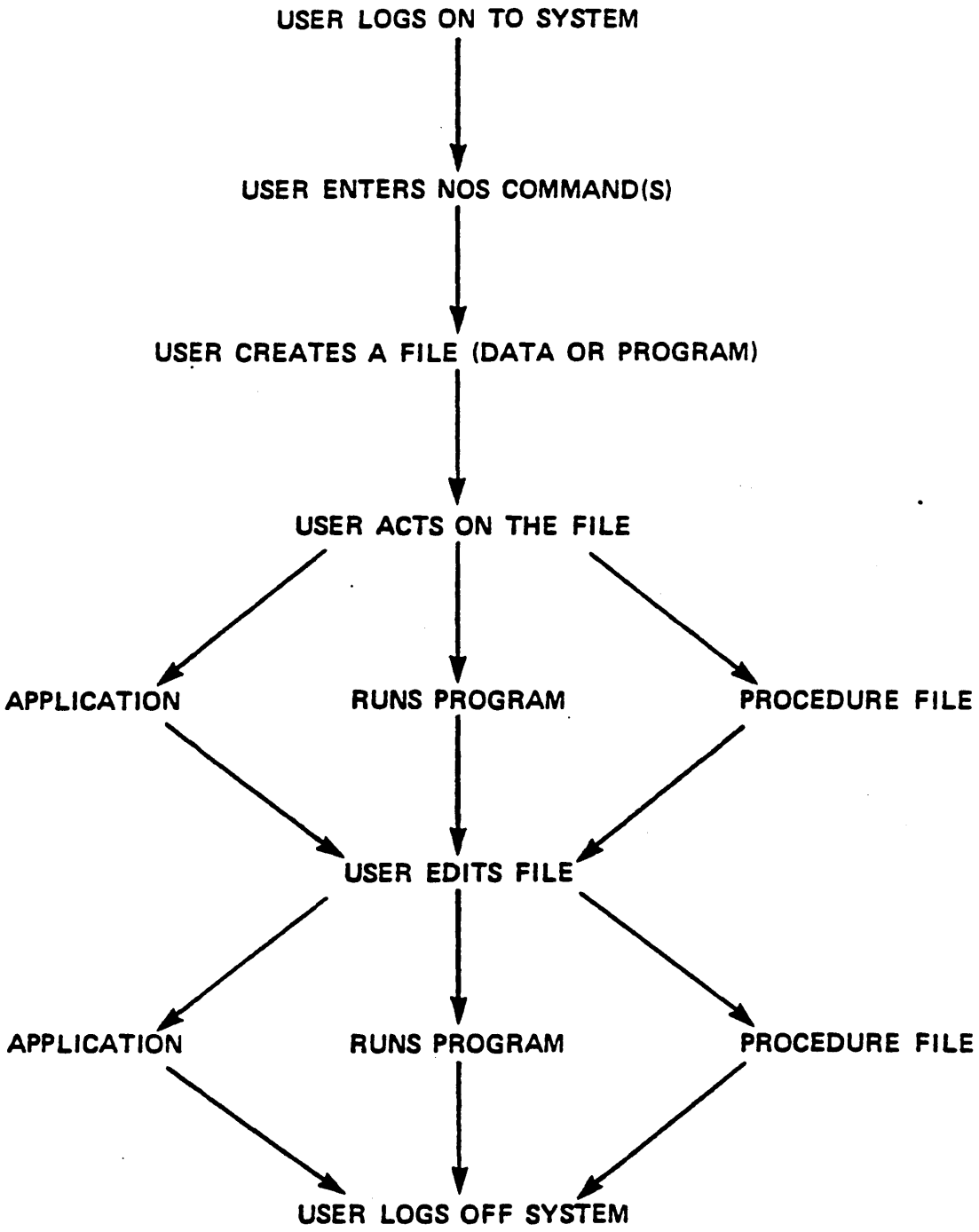
BATCH AND INTERACTIVE PROCESSING



Terminal Type		M33, M35, M37, M38	CDC 713–10
Parameter	Mnemonic		
Terminal class	TC	1	2
Control character	CT	ESC	ESCAPE
Backspace character	BS	CTRL H	←
Abort output line character	AL	CTRL X	CTRL X
User break 1 (interruption character)	B1	CTRL P	CTRL P
User break 2 (termination character)	B2	CTRL T	CTRL T

NOTE: CTRL H requires CTRL and H keys depressed simultaneously

DETACH COMMAND		ESC D	ESC D
IMMEDIATE JOB STATUS		ESC E	ESC E
ABBREVIATED JOB STATUS		ESC S	ESC S


USER LOGS ON TO SYSTEM
USER ENTERS NOS COMMAND(S)
USER CREATES A FILE (DATA OR PROGRAM)
USER ACTS ON THE FILE
APPLICATION
RUNS PROGRAM
PROCEDURE FILE
USER EDITS FILE
APPLICATION
RUNS PROGRAM
PROCEDURE FILE
USER LOGS OFF SYSTEM

LESSON 2

LOG-IN, LOG-OFF; TERMINAL CONTROL

LESSON PREVIEW:

This lesson presents log-on, log-off, terminal control and the creation and manipulation of local files, along with the declaration of subsystems and how to use them.

OBJECTIVES:

After completing this lesson the student should be able to:

- Log-on to system
- Subsystems
- ENQUIRE command

TRAINING AIDS:

None

PROJECTS:

Project 1

REFERENCES:

Interactive Facility Reference Manual, Chapters 2, 3, 4

LOG-IN PROCEDURE

1. DIAL NUMBER
2. WHEN CONNECTED:

   ENTER 2 CARRIAGE RETURNS
3. SYSTEMS RESPONDS FAMILY:

   ENTER FAMILY NAME
4. SYSTEM RESPONDS USERNAME:

   ENTER USERNAME
5. SYSTEM RESPONDS PASSWORD:

   ENTER PASSWORD
6. SYSTEM RESPONDS APPLICATION:

   ENTER IAF
7. SYSTEM RESPONDS CHARGE:

   ENTER CHARGE, CHARGE NUMBER, PROJECT NUMBER
8. SYSTEM RESPONDS READY

SESSION TERMINATION

WITH PHONE DISCONNECT

- LOGOUT
- BYE
- GOODBYE

WITHOUT PHONE DISCONNECT

- HELLO
- LOGIN

JOB SEQUENCE NAME (JSN)

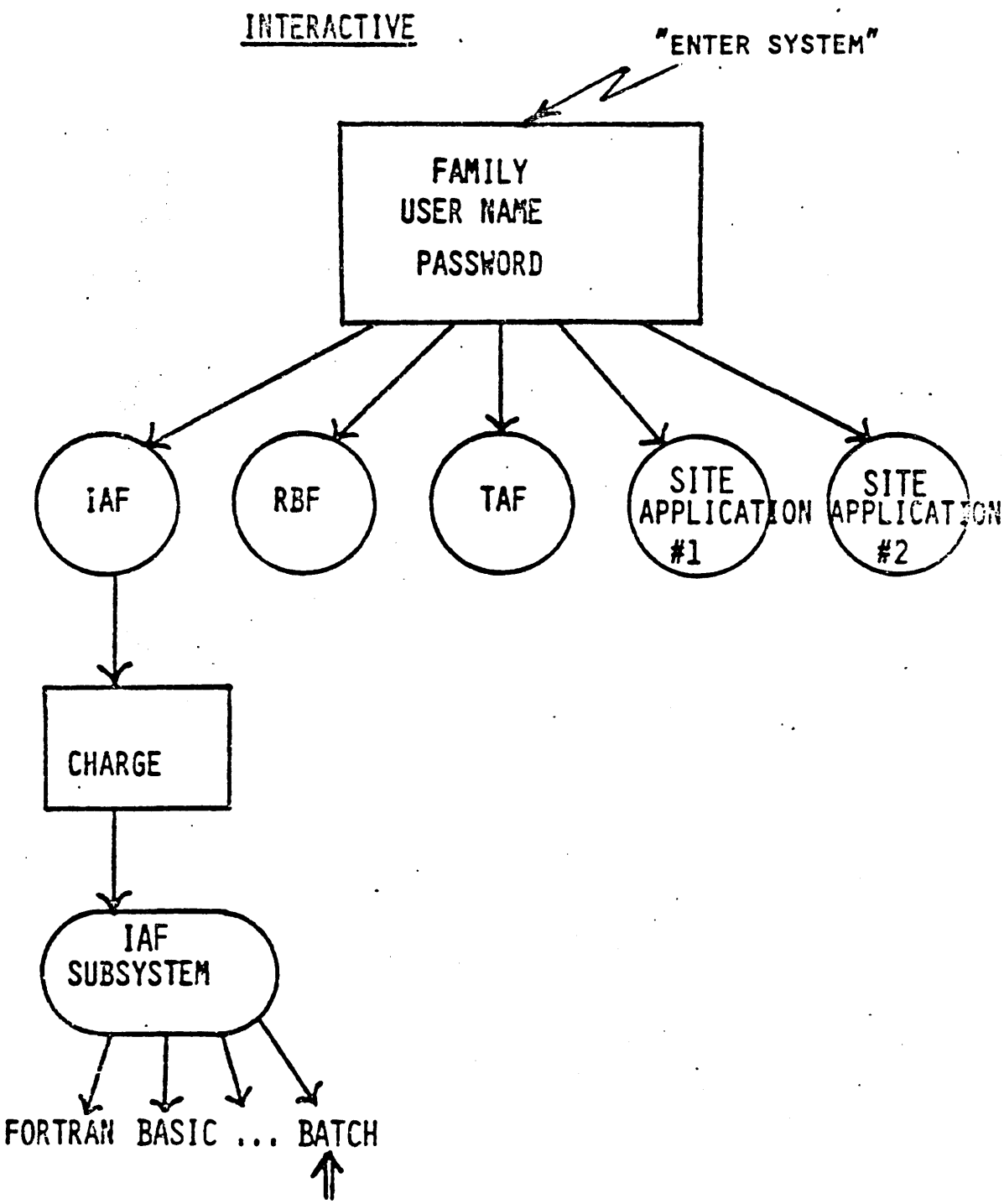
Every job and queued file is assigned a unique four character name defined by the system. The JSNs are the primary job identifiers for the system and user.

```
  WELCOME TO THE NOS SOFTWARE SYSTEM.

COPYRIGHT CONTROL DATA 1978, 1983.


84/02/21. 17.15.30. T01A61
(00) CY170-760 SN420. NOSCLSH.          NOS 23C9/6R2/R14T.
FAMILY:___

USER NAME: CPS4423

PASSWORD:  XXXX
  JSN: AEOR, NAMIAF
ITF IS NOW AFAILABLE.


LAST SYSNOTE UPDATE WAS 02/17/84
CLSH HOURS MON-THUR 0730-2200  FRI 0730-2000

$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
/LOGIN
UN=CPS4423  LOG OFF  17.16.20.
JSN=AEOR      SRU-S        2.082
IAF      CONNECT TIME 00.00.23.
 WELCOME TO THE NOS SOFTWARE SYSTEM.
COPYRIGHT CONTROL DATA 1978, 1983.


 84/02/21 17.16.21. T01A61
(00) CY170-760 SN420. NOSCLSH.          NOS 23C9/6R2/R14T.
FAMILY:     ,CPS4423

 ASSWORD:  XXXX

T01A61  - APPLICATION: IAF
JSN: AEOU, NAMIAF
```

```
LAST SYSNOTE UPDATE WAS 02/17/84
CLSH HOURS MON-THUR 0730-2200  FRI 0730-2000

$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
/HELLO
UN=CPS4423  LOG OFF  17.17.02.
JSN=AEOU      SRU-S        2.073
IAF      CONNECT TIME 00.00.17.
  WELCOME TO THE NOS SOFTWARE SYSTEM.
COPYRIGHT CONTROL DATA 1978, 1983.

 84/02/21 17.17.06. T01A61
(00) CY170-760 SN420. NOSCLSH.          NOS 23C9/6R2/R14T.
FAMILY:      ,CPS4423, XXXX

T01A61  - APPLICATION: IAF
JSN: AEOZ, NAMIAF
ITF IS NOW AFAILABLE.


LAST SYSNOTE UPDATE WAS 02/17/84
CLSH HOURS MON-THUR 0730-2200  FRI 0730-2000

$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
/BYE
 N=CPS4423  LOG OFF  17.17.49.
JSN=AEOZ      SRU-S        2.074
IAF      CONNECT TIME 00.00.19.
LOGGED OUT.
```

```
 WELCOME TO THE NOS SOFTWARE SYSTEM.
 COPYRIGHT CONTROL DATA 1978, 1983.

 84/02/21. 17.19.57. T01A61
 (00) CY170-760 SN420. NOSCLSH.          NOS 23C9/6R2/R14T.
 FAMILY:      ,CPS4423,XXXX
   JSN: AEPH, NAMIAF
 ITF IS NOW AFAILABLE.


 LAST SYSNOTE UPDATE WAS 02/17/84
 CLSH HOURS MON-THUR 0730-2200  FRI 0730-2000

 $REVERT,ABORT.* USER PROLOGUE NOT FOUND.
 $REVERT,ABORT.* USER PROLOGUE NOT FOUND.
 /NEW,FILE1
 /AUTO
 00100 THIS IS AN A EXAMPLE OF LOSING
 00110 A TELEPHONE LINE.......
 00120  PULL LINBE  E.....


   WELCOME TO THE NOS SOFTWARE SYSTEM.
 COPYRIGHT CONTROL DATA 1978, 1983.

 84/02/21. 17.22.42. T01A61
 (00) CY170-760 SN420. NOSCLSH.          NOS 23C9/6R2/R14T.
 FAMILY:      ,CPS4423,XXXX
   JSN: AEPV, NAMIAF
 ITF IS NOW AFAILABLE.
```

```
LAST SYSNOTE UPDATE WAS 02/17/84
CLSH HOURS MON-THUR 0730-2200  FRI 0730-2000

$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
/ENQUIRE,JSN

 JSN SC CS DS LID STATUS                    JSN SC CS DS LID STATUS

 AEPH.T.DT.BC.    .SUSPENDED                AEPV.T.ON.BC.    .EXECUTING
/RECOVER

   RECOVERABLE JOB(S)

   JSN     UJN         STATUS         TIMEOUT

   AEPH    ASEY        SUSPENDED      28 MIN.

ENTER  GO  TO CONTINUE CURRENT JOB,
       RELIST  TO LIST RECOVERABLE JOBS,
       OR DESIRED JSN:  AEPH
 JSN: AEPH  SYSTEM: BATCH     SRU:       2.067  FILE NAME: FILE1
 STATUS:
 CHARACTER SET: NORMAL   MODES: PROMPT ON
 IDLE.  ENTER COMMAND.
/LIST
00100 THIS IS AN EXAMPLE OF LOSING
00110 A TELEPHONE LINE.......
00120  PULL LINE.....
```



NOTE: THIS HAS NOTHING TO DO WITH REMOTE OR LOCAL BATCH, OR ANY MEANING OF THE ENGLISH WORD 'BATCH'.

IT SIMPLY 'OPENS THE DOOR' FOR A RICHER SET (THE FULL SET) OF NOS JCL STATEMENTS.

IAF SUBSYSTEMS

BATCH	All normal JCL commands available.
FTNTS	Special RUN command calls FTNTS. Limited other JCL. Abbreviations.
FORTRAN	Special RUN command calls FTN5. Limited other JCL. Abbreviations.
BASIC	Special RUN command calls BASIC. RESEQ automatically uses BASIC option. Limited other JCL. Abbreviations.
EXECUTE	User limited to execution of previously compiled programs and some other JCL. Abbreviations.
APL	Use APL interpreter.
NULL	Limited JCL. Abbreviations.

Sophisticated users will generally prefer the full features of the BATCH subsystem.

Beginning users may prefer FTNTS, FORTRAN, or BASIC, depending upon the language they are studying.

You may change subsystems at any time by typing the name of the subsystem you desire. Subsystem at sign-on is determined by your user number.

COMMANDS \ SUBSYSTEMS	FORTRAN	FTNTS	BASIC	EXECUTE	BATCH	NULL
RUN	compile and execute using FORTRAN 5 time share compiler	compile and execute using FORTRAN 4 time sharing compiler	compile and execute using time share BASIC compiler	execute previously compiled program	illegal command	illegal command
SAVE (primary file) †	tags primary file with FORTRAN flag	tags primary file with FTNTS flag	tags primary file with BASIC flag	tags primary file with EXECUTE flag	tags primary file with BATCH flag	no tag
LIST	no header	no header	no header	no header	no header	no header
LNH	no header	no header	no header	no header	no header	no header
RUN	no header	no header	no header	no header	illegal command	illegal command
RNH	no header	no header	no header	no header	illegal command	illegal command
RESEQ	no check for reference to line numbers in text	no check for reference to line numbers in text	check for references to line numbers in text	no check for reference to line numbers in text	no check for reference to line numbers in text	no check for reference to line numbers in text
Abbreviated commands	OK	OK	OK	OK	illegal command	OK

† If a file was saved with a subsystem flag, the terminal is put into that subsystem when the file is brought back as the primary file.

ENQUIRE

or

ENQUIRE, Options

A

B

D

F

J

L

R

S

T

U

JSN=

O =

FN=

USN=

```
/ATTACH,OPL/UN=LIBRARY
 ERROR IN ARGUMENT.
/ATTACH,OPL/UN=LIBRARY
/MODIFY,Z./*EDIT,CPUMTU R
ESC D

  JOB DETACHED, JSN=AEPH
ENQUIRE,JSN=AEPH
JSN= AEQV, NAMIAF
ITF IS NOW AFAILABLE.


LAST SYSNOTE UPDATE WAS 02/17/84
CLSH HOURS MON-THUR 0730-2200  FRI 0730-2000

$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
$REVERT,ABORT.* USER PROLOGUE NOT FOUND.
/
 AEPH.D.DT.BC.    .ROLLED                    UJN=ASEY
 SRUS=     12.239.  SRU LIMIT=NO LIMIT.  CM FL= 37600.  EXTENDE       OY
 ACSC,  D,      ,       8.766UNTS.

 DAYFILE=
   17.25.50.  $BEGIN(CATALOG,ZZZHELP,?)
   17.26.14.  $EXIT.
   17.26.19.CATLIST.
   17.26.33.CATLIST,LO=F.
   17.27.37.CATLIST,LO=F.
   17.28.43.ATTACH,OPL/UN=LIBRAR_Y.
   17.28.43. ERROR IN ARGUMENT.
   17.29.02.ATTACH,OPL/UN=LIBRARY.
   17.29.31.MODIFY,Z./*EDIT,CPUMTR
   17.29.58.ACSC,  D,      ,       8.766UNTS.
```

I. LOG-IN PROCEDURES

A. Establishing Communications

After you have applied power to the terminal and selected the proper mode, speed, and parity, you are ready to dial the terminal access number and make a connection between your terminal and the network.

The procedure used to make this connection depends upon the type of terminal and telephone/terminal interface being used to establish the connection. Two of the telephone/terminal interface devices which will be considered are the Data Set and the Acoustic Coupler.

Data Set - If a Data Set is being used and in not an integral part of the telephone:

1. Remove the telephone receiver from its cradle.

2. Press the "talk" button on the data set (telephone) to obtain a dial tone.

3. Dial the appropriate access telephone number that matchs your terminal's speed.

4. When the constant high-pitched carrier signal is heard it indicates that the call has been answered.

5. Press the data button on the data set, and place the receiver back in the cradle. Do not hang up the phone until you are done with the terminal. (If you are using a "cordless" or other special type telephone it may be necessary to not replace the receiver in the cradle, but to leave it "lay" removed from the telephone - preferrably where the "howl" of the carrier is not disrupting to the ears of nearby humans.)

If an acoustic coupler is being used:

1. Turn on the terminal and the acoustic coupler. If the coupler is included as a part of the telephone the dial tone will be heard when power is applied to the terminal. If not, the coupler must be turned on separately.

2. Dial the appropriate access number for your terminal speed.

3. When the constant high-pitched carrier signal is heard it indicates that the call has been answered.

4. Place the telephone receiver in the rubber suction cups of the acoustic coupler with the cord at the end indicated on the coupler. Make certain that the speaking end is facing in the direction indicated on the coupler.

B. Identify the Terminal Type

For asynchronous terminals simply press the carriage return. You have 60 seconds after connection is made to press the carriage return; otherwise the terminal will disconnect.

C. Specify the Family Name

After the terminal type has been identified, NOS indicates that access has been established by transmitting the current date and time of the connection, followed by the system header, and the family name request.

The family name identifies the family of permanent file devices. If your system has only one family, yours will be the default system, and you will need only to press the carriage return.

D. Enter User Name

After entering your family name, NOS will request that you supply your assigned User Name.

The User Name is an alphanumeric identifier assigned by the System Administrator when you were authorized to use NOS. This name uniquely identifies each user to the system and performs the following functions:

Controls access to the system
Defines privileges
Organizes the user's permanent files

E. Enter Your Password

After correctly entering your User Name, NOS continues the log-in procedure by requesting your password.

The password, in addition to the User Name, provides an additional level of security protection for the user. The selection of the password is completely under the control of the owner of the User Name, and can be changed whenever necessary by using the PASSWOR command.

F. Choose the Application

If your Family Name, User Name and Password combination is valid, NOS will request that you enter an Application name.

LESSON 3

NOS LINE AND FILE CONCEPTS

LESSON PREVIEW:

This lesson presents line and file concepts and explains terminal input.

OBJECTIVES:

After completing this lesson the student should be able to:

- Explain the concepts of record and file
- Be able to identify information the system accepts as data and information it accepts as commands
- Create a local file
- List local files
- Manipulate local files
  - new
  - LIST
  - SORT
  - NOSORT
  - RESEQ
  - TEXT

TRAINING AIDS:

Visuals

PROJECTS:

None

REFERENCES:

Interactive Facility Reference Manual

TTY USER FILES

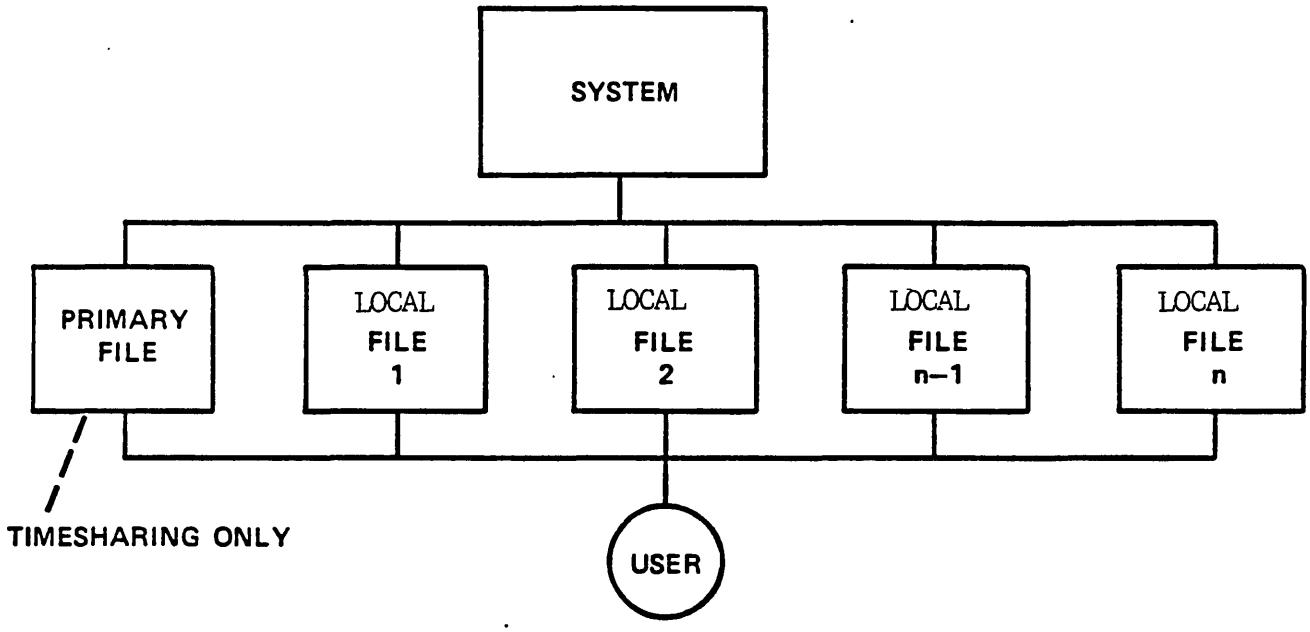
LOCAL FILE - Any file assigned to a user while he is logged in to a terminal.

PRIMARY FILE - Any local file retrieved from the Permanent File System with the OLD or LIB command or initiated with the NEW command. The primary file is assumed to be the file on which most operations are performed unless another file is specified. The primary file is automatically sorted prior to each use and rewound after each use.

Note: There can be only one PRIMARY file assigned to each user at any one time. Execution of a NEW, OLD, or LIB command will automatically drop the current PRIMARY file and 'return' all other working files to the system. The 'return' of the non-primary files can be inhibited by using the ND parameter on the OLD, NEW, or LIB command.

Every time a LIST or LNH is executed, a REWIND is done first automatically, regardless of whether the file is primary or non-primary.

Note: A user can only manipulate local files. At the termination of a job, all local files are return (not save!)



UNDER NOS MULTIPLE LOCAL FILES MAY BE ASSOCIATED WITH THE USER'S JOB AT ANY GIVEN POINT IN TIME. THESE LOCAL FILES MAY CONTAIN PROGRAMS, DATA, TEXT, OR MERELY BE SOURCE FILES. UNDER TIMESHARING THERE ALSO IS DEFINED A SPECIAL LOCAL FILE THAT IS DESIGNATED AS THE PRIMARY FILE. THIS PRIMARY FILE HAS PARTICULAR SIGNIFICANCE IN THAT MANY OPERATIONS SUCH AS SORTING, PACKING, AND REWINDING WILL BE PERFORMED AUTOMATICALLY BY THE SYSTEM UNLESS THE USER SPECIFIES OTHERWISE.

LOCAL FILE RULES

1. Any temporary file is considered to be a local file.

2. Local files may contain programs, data, scratch garbage, etc.

3. Local files remain:
   a) With the timesharing user until log-off.
   b) With the batch user until job completion or
   c) Until released with the RETURN command/control statement.
   d) The CLEAR command instantly releases all local files.

4. There are no operations automatically performed on non-primary local files, such as REWINDs, PACKs, or SORTs. An exception is that a REWIND will be automatically done on a non-primary local file when a list, F= or LNH, F= is executed. Whenever an operation is performed using a non-primary local file, particularly under timesharing, it should be rewound, unless some special positioning is desired by the user.

5. Local files may be added to an already existing permanent file by using the APPEND command/control statement.

6. A NEW, OLD, or LIB command drops all local files unless the NODROP parameter is used. Doesn't apply to LIBRARY under batch processing.

TERMINAL INPUT

NORMAL

DATA LINES BEGIN WITH A NUMERIC.
ALL OTHER LINES ARE INTERPRETED
AS COMMANDS.

TEXT

ALL LINES ARE DATA.

ALL DATA LINES ARE WRITTEN TO
THE PRIMARY FILE.

GENERAL COMMANDS

- NEW
- AUTO
- RESEQ
- LIST
- LNH
- PACK
- SORT
- NOSORT
- TEXT
- PRIMARY
- OLD

```
NEW,AAGGG

 READY.
AUTO
00100 AAAA
00110 BBBB
00120 CCCC
00130                    (CNTL) (X)
◆DEL◆
LIST                     ENTERING A COMMAND TAKES USER OUT OF 'AUTO-MODE'

00100 AAAA
00110 BBBB
00120 CCCC

 READY.
NEW,AAAGG

READY.
AUTO,200,100
00200 AAAA
00300 BBBB
00400 CCCC
00500 DDDD               (CNTL) (X)
◆DEL◆
00525 DDDD
00625 EEEE
00725                    (CNTL) (X)
◆DEL◆
LIST

00200 AAAA
00300 BBBB
00400 CCCC
00525 DDDD
00625 EEEE

 READY.
525 I WANT TO CHANGE THIS LINE
LIST
00200 AAAA
00300 BBBB
00400 CCCC
525 I WANT TO CHANGE THIS LINE
00625 EEEE

 READY.
```

```
NEW,AAGGG

 READY.
AUTO,1000,25
01000 AAAA
01025 BBBB
01050 CCCC
01075 <--- (CR)
01100 <--- (CR)
01125 FFFF
01150 <--- (CNTL) (X)
◆DEL◆
10 THIS IS THE 1ST NEW LINE
00035 THIS IS THE 2ND NEW LINE
00060 <--- (CNTL) (X)
◆DEL◆
01150 JJJJ
01175 KKKK
01200 <--- (CNTL) (X)
◆DEL◆
LIST <--- "COMMANDS" TAKE USER OUT OF "AUTO-MODE"

10 THIS IS THE 1ST NEW LINE
00035 THIS IS THE 2ND NEW LINE
01000 AAAA
01025 BBBB
01050 CCCC
01125 FFFF
01150 JJJJ
01175 KKKK

 READY.
```

AUTOMATIC SORT DONE WHEN NEXT:

LIST	LENGTH
LNH	REPLACE
RUN	SAVE
RNH	SUBMIT ,
EDIT	
XEDIT ,	

IS DONE. THESE COMMANDS SET THE 'SORT-FLAG.'

```
 READY.
AUTO
00100 AAA
00110 BBB
00120 CCC
00130 DDD
00140 <-------- (CNTL) (X)
◆DEL◆
00110 THIS IS 1ST LINE SORTED IN
00120 <-------- (CNTL) (X)
◆DEL◆
00115 THIS IS 2ND LINE SORTED IN
00125 <-------- (CNTL) (X)
◆DEL◆
NOSORT
 READY.
00102 THIS IS 3RD LINE
00103 THIS IS 4TH LINE
00104 THIS IS 5TH LINE
LIST

00100 AAA
00102 THIS IS 3RD LINE
00103 THIS IS 4TH LINE
00104 THIS IS 5TH LINE
00110 THIS IS 1ST LINE SORTED IN
00115 THIS IS 2ND LILNE SORTED IN
00120 CCC
00130 DDD

 READY.
```

NOSORT REMAINS IN EFFECT ONLY UNTIL NEXT NUMBERED LINE IS ENTERED.
ALL SORT and NOSORT DO IS SET OR TAKE OFF 'SORT-FLAG.'

```
 READY.
TEXT
ENTER TEXT MODE.

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
(CTRL) (P)
 EXIT TEXT MODE.

 READY.
SAVE

 READY.
LIST

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
 READY.
```

```
ENQUIRE
TERMINAL:     22, NAMIAF
SYSTEM:   BASIC
FILE NAME:
STATUS:   IDLE
MESSAGE:
GET,COWBOY1

 READY.
ENQUIRE
TERMINAL:     22, NAMIAF
SYSTEM:   BASIC
FILE NAME:
STATUS:   IDLE
MESSAGE:
ENQUIRE,F

LOCAL FILE INFORMATION.

 FILENAME  LENGTH/PRUS  TYPE  STATUS

  INPUT*                IN.*  EOF WRITE
  COWBOY1          1    LO.   BOI READ
  OUTPUT                LO.   I/C WRITE

 TOTAL = 3

 READY.
OLD,SUSAN/ND

 READY.
ENQUIRE
TERMINAL:     22, NAMIAF
SYSTEM:   BASIC
FILE NAME: SUSAN
STATUS:   IDLE
MESSAGE:
ENQUIRE,F

LOCAL FILE INFORMATION.

 FILENAME  LENGTH/PRUS  TYPE  STATUS

  INPUT*                IN.*  EOF WRITE
  COWBOY1          1    LO.   BOI READ
  OUTPUT                LO.   I/C WRITE
  SUSAN            1    PT.   BOI READ

 TOTAL = 4

 READY.
ENQUIRE,T

CPU ACCUMULATOR.

 CPU TIME     0.025 SECS.

READY.
```

```
 READY.
LIST
00100 PRINT   "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00150 PRINT   "OR ENTER CTRL/T TO TERMINATE THE PROGRAM"
00200 INPUT   X,Y,Z
00250 LET A = X+Y+Z
00300 PRINT   "X+Y+Z="A
00350 LET B = X*Y
00400 PRINT   "X*Y="B
00450 LET C   A+B
00050 PRINT   "A+B="C
00550 GO TO 002000
00600 END

 READY.
BATCH
RFL,0.
/BASIC,I=SUSAN,B=SAMMY
ILLEGAL COMMAND.
ENQUIRE
TERMINAL:     22, NAMIAF
SYSTEM:   BASIC
FILE NAME: SUSAN
STATUS:   IDLE
MESSAGE:
BATCH
$RFL,0.
/X,BASIC,I=SUSAN,B=SAMMY
      .008 CP SECONDS COMPILATION TIME
/REWIND,SAMMY
$REWIND,SAMMY.
/TDUMP,I=SAMMY
 F  1 R   1 W     0- 7700 0016 0000 0000 0000  2325 2301 1600 0000 0000

                     ;      N                     S U  S A  N
 F  1 R   1 W     2- 4334 5034 3550 3336 5755  3436 5735 4257 3340 5755

                    81   / 1  2 /  0 3  .     1 3  . 2  7 .  0 5  .
 F  1 R   1 W     4- 5516 1723 5534 5737 5555  0201 2311 0355 5536 5740

                       N  O S    1  . 4        B A  S I  C      3  . 5

 F  1 R   1 W     6- 4334 3542 3455 5555 5555  5555 5555 5555 5555 5555

                    8 1  2 7  1
 F  1 R   1 W    10- 5555 5555 5555 5555 5555  5555 5555 5555 5555 5555

 -- ABOVE LINE REPEATED --
 F  1 R   1 W    16- 5555 5555 5555 5555 5555
 *TERMINATED*

 F  1 R   1 W    10- 5555 5555 5555 5555 5555  5555 5555 5555 5555 5555

 -- ABOVE LINE REPEATED --
 F  1 R   1 W    16- 5555 5555 5555 5555 5555
 *TERMINATED*
```

```
LNH,F=SUSAN
00100 PRINT   "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00150 PRINT   "OR ENTER CTRL  TP
00150 PRINT   "OR ENTER CTRL/T TO TERMINATE THE PROGRAM"
00200 INPUT   X,Y,Z
00250 LET A = X+Y+Z
00300 PRINT   "X+Y+Z="A
00350 LET B = X*Y
00400 PRINT   "X*Y="B
00450 LET C   A+B
00500 PRINT   "A+B="C
00550 GO TO 00200
00600 END

 READY.
ILLEGAL COMMAND
BASIC
 OLD, NEW, OR LIB FILE:  RUN,I=SUSAN
   INPUT FILE EMPTY OR MISPOSITIONED

 RUN COMPLETE.
REWIND,SUSAN
 READY.
RUN,I=SUSAN
ENTER THREE NUMBERS IN THE FORM X,Y,Z
OR ENTER CTRL/T TO TERMINATE THE PROGRAM
? 1,2,3,5
X+Y+Z= 6.5
X*Y= 2
A+B= 8.5
?*CTRL T
 *TERMINATED*
LNH,F=SUSAN

00100 PRINT       "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00150 PRINT       "OR ENTER CTR:
* INTERRUPTED *

IDLE
```

(FIRST, CTRL P , WITH NO CR TO INTERRUPT;
SECOND, P AND CR TO CONTINUE.)

FIRST, CTRL P , WITH NO CR TO INTERRUPT;
SECOND, CR AND CTRL T TO TERMINATE.

```
GET,SUSAN
 READY.
BATCH
RFL,0.
/X,BASIC,I=SUSAN,B=BINARY
       .088 CP SECONDS COMPILATION TIME
/EXECUTE
OLD, NEW, OR LIB FILE:  RUN,I=BINARY
ENTER THREE NUMBERS IN THE FORM X,Y,Z
OR ENTER CTRL/T TO TERMINATE THE PROGRAM
? 1.2,112,8
X+Y+Z= 121.2
X*Y= 134.4
A+B= 255.6
?
 *TERMINATED*

RNH,I=BINARY
ENTER THREE NUMBERS IN THE FORM X,Y,Z
OR ENTER CTRL/T TO TERMINATE THE PROGRAM
? 1,2,3
X+Y+Z= 6
X*Y= 2
A+B= 8
?
 *TERMINATED*

FTNTS
 OLD, NEW, OR LIB FILE:RNH, I=BINARY


 - * TRIVIAL PROGRAM UNIT IGNORED
EMPTY LOAD FILE  -  LGO

 RUN COMPLETE.
BASIC
 OLD, NEW, OR LIB FILE:EXECUTE
 OLD, NEW, OR LIB FILE:RNH, I=BINARY
ENTER THREE NUMBER IN THE FORM X,Y,Z
OR ENTER CTRL/T TO TERMINATE THE PROGRAM

? 2,6.4,1.0
X+Y+Z= 9.4
X*Y= 12.8
A+B= 22.2
?

 *TERMINATED*
HELLO
UN=ITU4063    LOG OFF              11.21.56.
JSN=ABAL      SRU       2.816        UNITS.
IAF      CONNECT TIME 00.02.42.
```

LOCAL FILES

	PRIMARY	NON-PRIMARY
CREATE A LOCAL FILE	NEW, FILENM	
LIST A LOCAL FILE	LIST OR LNH	LIST, F=FILENM OR LNH, F=FILENM
COMPILE &/OR EXECUTE	RUN OR RNH	RUN, I=FILENM OR RNH, I=FILENM
ENTER 'TEXT' MODE	TEXT	
EXIT 'TEXT' MODE	(CTRL) (P)	
SAVE A LOCAL FILE AS A PERM. FILE	SAVE	SAVE, FILENM
RETRIEVE A COPY OF PERM. FILE AS A LOCAL FILE	OLD, FILENM	GET, FILENM
SORT	(DONE AUTOMATICALLY)	SORT, FILENM
REWIND TO BOI	(DONE AUTOMATICALLY)	REWIND, FILENM
ELIMINATE -EOR-	PACK	PACK, FILENM

PROJECT NO. 1

CYBER NOS TTY USAGE

1. Log-in with User Number assigned by instructor.

2. Enter the following BASIC program as a NEW file. Have the system number your lines. Use 1000 as the first line number and make the increment 100.

```
1000 INPUT   X,Y,Z
1100 LET A = X+Y+Z
1200 PRINT   "X+Y+Z="A
1300 LET B = X*Y
1400 PRINT   "X*Y="B
1500 LET C = A+B
1600 PRINT   "A+B="C
1700 GO TO   1000
1800 END
```

3. List the file.

4. Compile and execute the program.

5. If necessary, make corrections to the program and repeat Step 4 until it works.

6. Add the following lines to the source program:

```
500 PRINT "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
600 PRINT "OR ENTER (CTRL/T) TO TERMINATE THE PROGRAM"
```

7. Compile and execute again.

8. When it works properly, SAVE the source file as a permanent file.

9. Compile the program, placing the binary object program on a file.

10. Execute the binary program.

11. When it works properly, SAVE the binary program as a permanent file.

12. Retrieve the source file as your primary file.

13. Compile and execute it.

14. Retrieve the source file as a non-primary working file called X.

15. Compile and execute it.

16. Retrieve the source file as your primary file.

17. Execute it.

18. Retrieve the binary program as a non-primary working file called Y.

19. Execute it.

LESSON 4

PERMANENT FILES

LESSON PREVIEW:

This lesson presents permanent files, introduces indirect access and direct access permanent file concepts and associated commands.

OBJECTIVES:

After completing this lesson the student should be able to:

- Explain the concept of indirect access permanent files and list the applicable commands (SAVE, GET, OLD, REPLACE)
- Explain the concept of direct access permanent files and list the appropriate commands (DEFINE, ATTACH)
- List other permanent file commands and describe what they do (CATLIST, PURGE, PERMIT, APPEND, CHANGE)

TRAINING AIDS:

Visuals

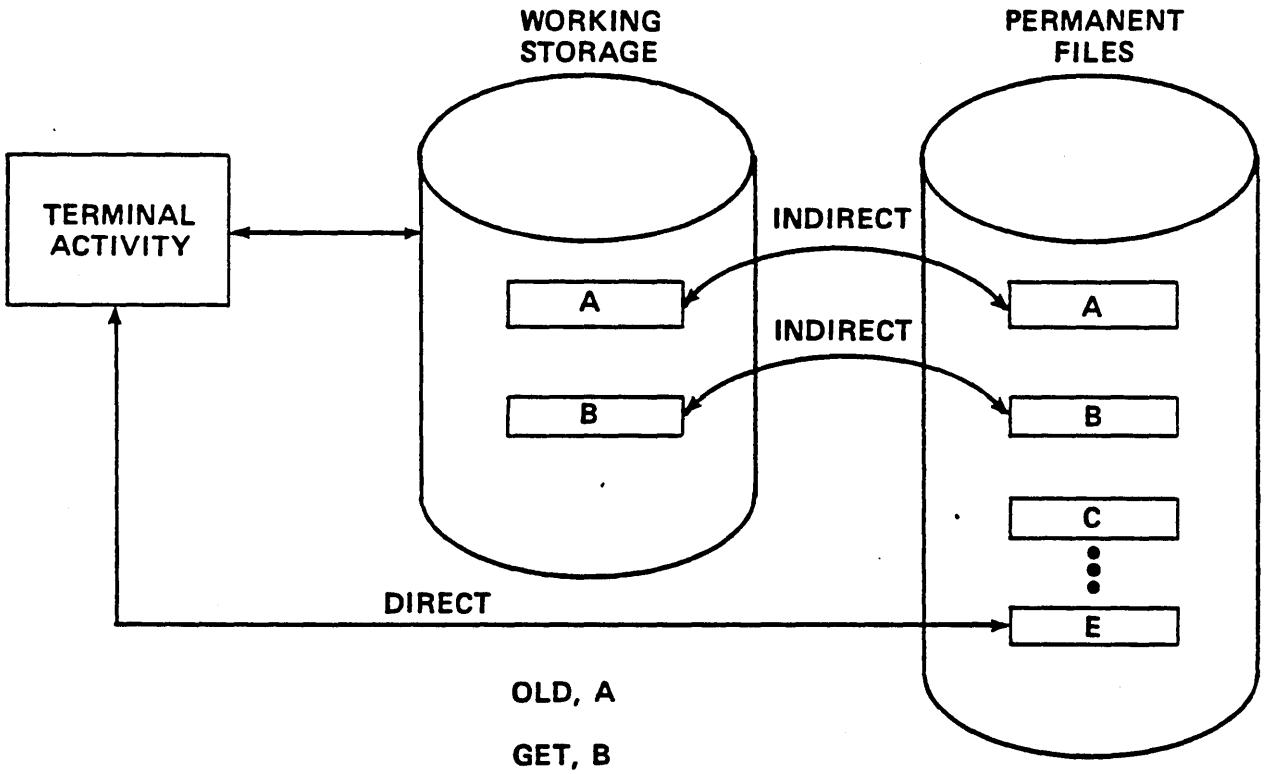
PROJECTS:

Project 2
Exercise 1

REFERENCES:

Interactive Facility Reference Manual, Chapters 5 and 6

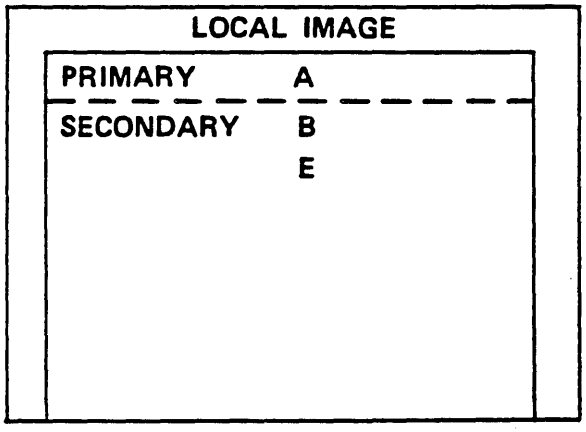
FILE CONCEPTS



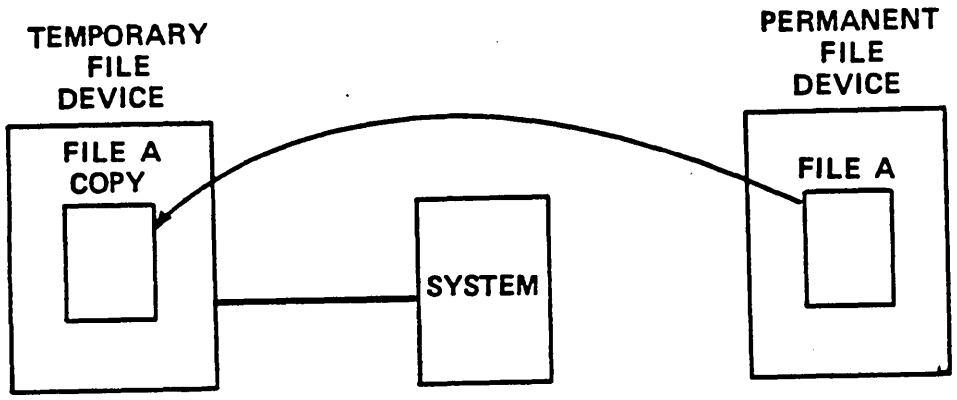
OLD, A

GET, B

ATTACH, E

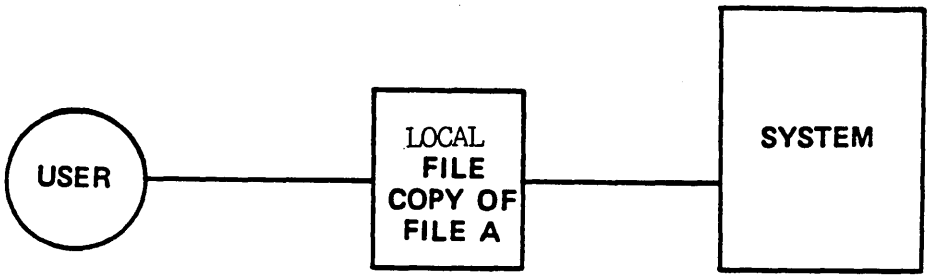


INDIRECT ACCESS - PERMANENT FILES



METHOD OF INDIRECT ACCESS

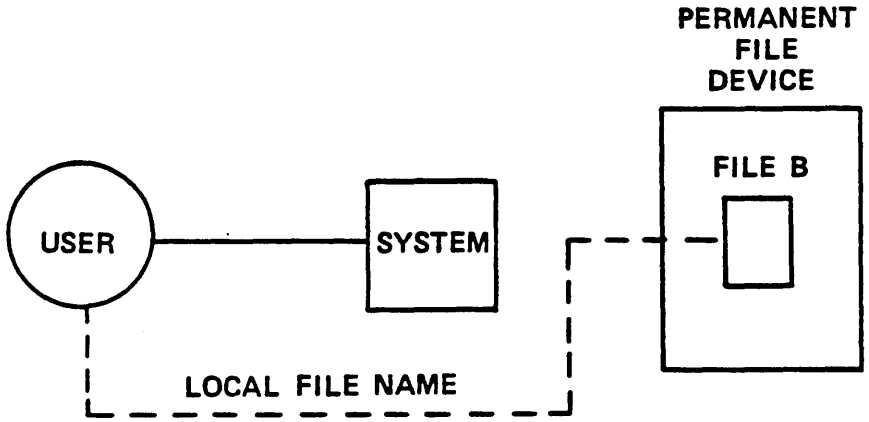
The only time that the system knows there is any link between file A and its copy is when the actual copy is performed. After the copy is complete, the system has no idea that there was any connection between the original file and its copy.



AFTER COPY MADE

The inherent advantage here is that regardless of what is done to the content of the local file, the permanent file remains unaltered and intact. One cannot accidentally damage the contents of the permanent file.

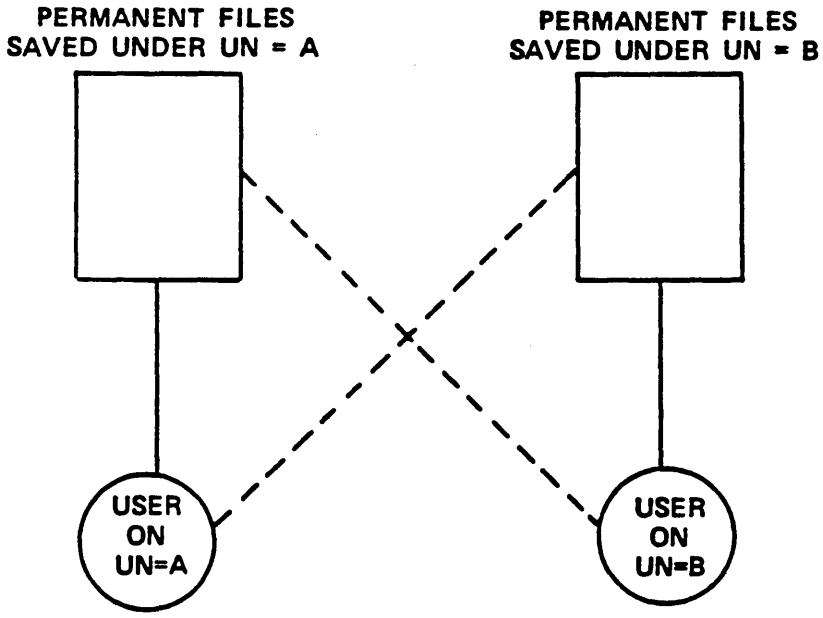
DIRECT ACCESS - PERMANENT FILES



METHOD OF DIRECT ACCESS

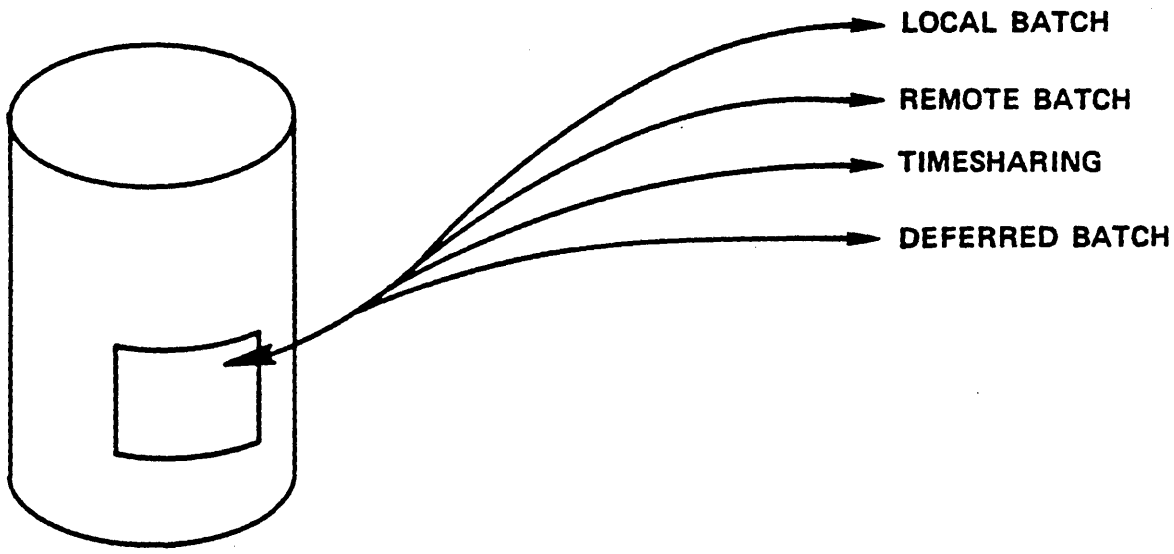
No copy is made of the direct access permanent file. The permanent file is attached directly to the user, referenced by a local file name. At job termination the link between the user and the direct access permanent file is destroyed, with the permanent file remaining in the same state it was at job termination time. While the file is attached to the user's job, any alteration made will be permanent, with no method of recovery of the original content of the file. Thus one must use extreme caution to avoid destroying important areas of the file accidentally.

PERMANENT FILE SECURITY

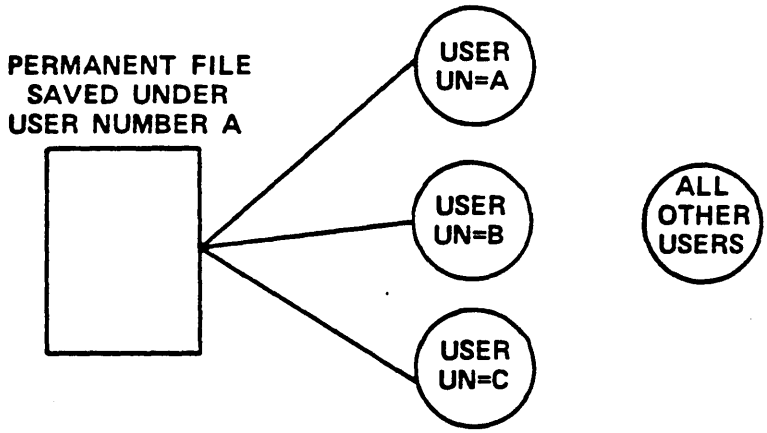


Permanent file protection only on dotted line accesses. Anyone logging on a given user number may purge any file on that same user number regardless of access privileges defined for that file.

FILE COMPATIBILTY



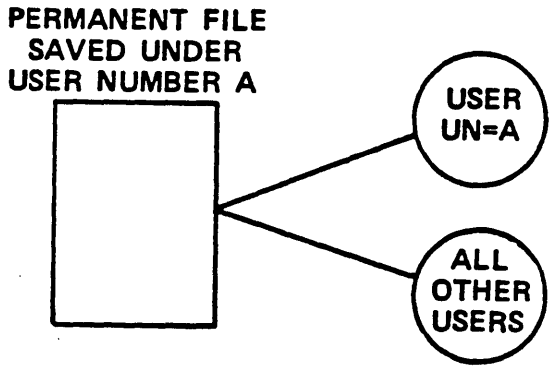
PRIVATE FILE ACCESSING



User number A has access to the file, since the file was saved under his user number. User number A may execute, read, append or write on the file regardless of the access privileges defined when the file was saved. Any user who may perform the same operations on the file as user A is said to have ultimate privileges.

In this case users B & C have been given explicit permission to access the file by user A. In order to accomplish this, user A used either A PERMIT command or control statement. Only users given explicit permission in the permit operation are allowed access to a private permanent file. Even though they have explicit permission, however, users B & C must still know that the file is saved under user number A and the file password, if any, which A has placed on the file in order to actually access it.

PERMISSION CATEGORIES - SEMIPRIVATE AND PUBLIC



SEMIPRIVATE OR PUBLIC FILE ACCESSING

Again, user number"A" has ultimate privileges since the file was created under his user number. All others may access the file, provided they know the relevant protection information, i.e., the user number A that the file was created under and the file password, if any. If the file is categorized as semiprivate, full bookkeeping as to user numbers who have accessed the file, is recorded.

PERMANENT FILES

INDIRECT ACCESS	ACTION
SAVE	CREATE INDIRECT ACCESS PERMANENT FILE
OLD, LIB	RETRIEVE INDIRECT FILE AS PRIMARY FILE
GET	RETRIEVE INDIRECT FILE AS SECONDARY FILE
PURGE	REMOVES PERMANENT FILE CATALOG ENTRY
APPEND	ADDS INFORMATION TO PERMANENT FILE AT END-OF-INFORMATION
REPLACE	REPLACE INDIRECT PERMANENT FILE WITH A NEW COPY
PERMIT	GRANT ACCESS PERMISSION
CATLIST	OBTAIN INFORMATION CONCERNING PERMANENT FILE
RETURN	RELEASES SPECIFIC WORKING FILE
CLEAR	RELEASE ALL WORKING FILE
(LOG-OFF)	RELEASE ALL WORKING FILE
CHANGE OF PRIMARY FILE	POSSIBLE RELEASE OF ALL WORKING FILE

DIRECT ACCESS	ACTION
DEFINE	CREATE DIRECT ACCESS PERMANENT FILE
ATTACH	ACCESS DIRECT ACCESS FILE
PURGE	REMOVE PERMANENT FILE CATALOG ENTRY

LOCAL FILES

LIST	REWINDS AND LISTS CONTENTS OF FILE
LNH	SAME AS 'LIST',
NEW	CREATE 'PRIMARY' LOCAL FILE
RUN	COMPILE AND/OR EXECUTE FILE
RNH	SAME AS 'RUN',
ENQUIRE	OBTAIN INFORMATION ABOUT TERMINAL/FILES

USER "SMS2005" LOGS IN

SAVE	ABC = XYZ/ (lfn) (pfn)	PW = CAT, (optional; not same as login)	CT = P, ('P' by default; or S or PU)	M = W ('W' by default)

1) Private - a) Alt. users must be granted specific access permission via a PERMIT.
   b) Originator may find out who accessed his files via CATLIST.

2) Semi-Private - a) Alt. users can access file without PERMIT.
   b) Originator may find out who accessed his files.

3) Public - a) Alt. users can access file without PERMIT.
   b) Originator cannot find out who accessed his file.

PERMIT, XYZ, SMS25 = R

USER "SMS25" LOGS IN

OLD, MNO=XYZ/UN =SMS2005, PW=CAT

"SMS25," GETS FILE IN 'READ - MODE' ONLY.

PERMIT, pfn, un = perm. mode

1) Can be used to allow alt.-users to gain certain accesses to private AND semi-private files.

   a) Necessary for private files.
   b) Not necessary for semi-private files.
      (Used to limit alternate user access to permanent file)

2) If perm. mode is missing; "un=R" by default.
   (alternate user permission is READ)

3) Ex - SAVE, ABC/CT=S
   PERMIT, ABC, SMS25 = R

   If above PERMIT missing, SMS25 could get ABC with write-permission

To Rescind previously granted PERMIT

1) Remove PERMIT information
   OLD, SUSAN
   PURGE, SUSAN
   SAVE, SUSAN

2) Change perm. file name. CHANGE, NEWFILE=OLD FILE

3) Change or create a password. CHANGE, SUSAN/PW=CAT

4) Remove Access. PERMIT, XYZ, SMS25=NULL

CATLIST, LO=FP

1) For PRIVATE and SEMI-PRIVATE files only.

2) Specifies who has been granted access with a PERMIT; specified by an *.

3) Gives list of alt. users accessing files; include number of accesses, date and time of last access.

CATLIST - Gives name of all of your permanent files.

CATLIST,LO=F - Gives full list options on all your permanent files.

CATLIST,LO=F, FN=XYZ - Gives full list options on only your file 'XYZ'.

CATLIST,LO=F, UN=SMS25 - Gives full list options on all files in SMS25's catalog that YOU MAY ACCESS. Will NOT list passwords associated with the files.

```
CATLIST,LO=F

CATALOG OF  CPS4423          FM/NOSCLSH 84/02/23. 17.41.52.

    FILE NAME  FILE TYPE        LENGTH IN CREATION  ACCESS     DATA MOD
     PASSWORD  COUNT INDEX  PERM. SUBSYS DATE/TIME DATE/TIME DATE/TIME
      EXPIRES  LEVEL  PR BR RS

    1 MAT      DIR. PRIVATE          0  * 84/02/23. 84/02/23. 84/02/23.
      WEWEWE        0           READ      17.37.40. 17.37.40. 17.37.40.
                          N  Y  D

    2 MARY     DIR. PUBLIC           0  * 84/02/23. 84/02/23. 84/02/23.
      MARYONE       0           READ      17.39.59. 17.39.59. 17.39.59.
                          N  Y  D

    3 SAMPLE   IND. PRIVATE        115    84/01/11. 84/01/16. 84/01/11.
                    3           WRITE     15.53.34. 09.15.14. 15.55.51.
                          N  Y  D

    4 DU3A     IND. PRIVATE        165    84/01/30. 84/01/31. 84/01/30.
                    1           READ      12.37.25. 08.58.55. 12.37.25.
                          N  Y  D

    5 DU3B     IND. PRIVATE         92    84/01/30. 84/01/31. 84/01/30.
                    1           READ      12.37.26. 13.10.16. 12.37.26.
                          N  Y  D

    6 DU1A     IND. PRIVATE        131    84/01/30. 84/01/30. 84/01/30.
                    1           READ      12.40.36. 13.36.41. 12.40.36.
                          N  Y  D
```

```
/CATLIST,UN=LIBRARY
 CATALOG OF  CPS4423          FM/NOSCLSH 84/02/23. 17.42.39.
          ALTERNATE CATALOG   LIBRARY


 INDIRECT ACCESS FILES

  BULLET    FSTEACH   MESSG     PROC2     RHFINFO   SYMPCOD   SYSHIST
  DFFIDOC   HEADING   PASWORD   RBFBULL   RHFNOTE   SYSBULL   SYSNOTE
  FORM      JOBSLIP   PHONES    RECLAIM   RHFPRNT   SYSBUL1   USABIN
  FSEBUG    MAILBOX   PROCS     RHFBULL   R6NOTE    SYSBUL2   XEDITH
  FSEPROC

 DIRECT ACCESS FILES

  AP1LIB1   C ** S    OM
PROC   FTN5      MASSEM    OLMLIB    RMFIO
  AP1LIB2   CMACRO    FSEHELP   HELPLIB   MPASCAL   OPASLIB   SMAC17
  AP1LIB3   CMDFILE   FSGUIDE   HLPFILE   MPEDIT    OPL       TOTAL
  ASSEM     CODING    FTNLIB    HOURLY    OHLPLIB   P*8P      +5
OPY
  BECMDFI

        29 INDIRECT ACCESS FILES,     TOTAL PRUS =      931.
        29 DIRECT ACCESS FILES,       TOTAL PRUS =    74877.
```

```
CATLIST,LO=F,UN=LIBRARY,FN=BULLET
CATALOG OF  CPS4423          FM/NOSCLSH 84/02/23. 17.43.29.
          ALTERNATE CATALOG   LIBRARY
   FILE NAME  FILE TYPE       LENGTH IN CREATION  ACCESS     DATA MOD
              MD/CNT INDEX--PERM- SUBSYS DATE/TIME DATE/TIME DATE/TIME
     EXPIRES   LEVEL  PR BR RS
 1 BULLET   IND. SEMI-PR      2     81/07/21. 84/02/16. 81/12/11.
                758           READ       10.18.53. 11.57.43. 14.47.39.
                          N  Y  D
       1 INDIRECT ACCESS FILE,      TOTAL PRUS =        2.
,CATLIST,FN=BULLET,UN=LIBRARY,LO=FP
 BULLET NOT FOUND.
/CATLIST,FN=MARY,LO=FP
 NO PERMITS.


BYE
UN=CPS4423  LOG OFF  17.45.38.
JSN=AEFB    SRU-S    2.525
IAF     CONNECT TIME 00.11.03.
```

PERMANENT FILE SUMMARY

INDIRECT ACCESS	DIRECT ACCESS
1) WORK WITH A COPY.	1) WORK WITH FILE.
2) CREATED BY SAVE.	2) CREATED BY DEFINE.
3) ACCESSED BY OLD, LIBRARY, GET.	3) ACCESSED BY ATTACH.
4) SPACE ALLOCATED BY 'SECTOR'	4) SPACE ALLOCATED BY 'TRACK'
5) NO 'WRITE-INTERLOCK'	5) IF ONE USER HAS IT ATTACHED IN WRITE-MOD:, ANOTHER USER CANNOT ATTACH IT.
6) WHEN PRIMARY FILE IS MADE PERMANENT, THE SUBSYSTEM IS RETAINED.	6) YOU CANNOT SAVE A SUB-SYSTEM WITH A DIRECT ACCESS FILE.
7) MAY BE BUILT THROUGH USE OF AUTO, TEXT.	7) YOU CANNOT USE TEXT OR AUTO TO BUILD A DIRECT ACCESS FILE.
8) FILE IS CREATED IN WRITE-MODE BY DEFAULT.	8) FILE IS CREATED IN WRITE-MODE BY DEFAULT.
9) ORIGINATOR GETS FILE IN WRITE-MODE BY DEFAULT.	9) ORIGINATOR GETS FILE IN READ-MODE BY DEFAULT.

PROJECT NO. 2

CYBER NOS TTY USAGE

1. Log-in.
2. Create a file.
3. List it.
4. Save it as an Indirect Access, Semi-Private permanent file on which an alternate user will have Read-Only permission. Require a password to be given when accessing the file. (Instructor will assign Alternate User Number.)
5. Get a full catalog listing for that file only.
6. Create another file.
7. List it.
8. Save it as an Indirect Access, Private permanent file.
9. Give alternate User "WRITE" permission on the file.
10. Log-in under alternate User number.
11. Retrieve both files created above and list them.
12. Create a file.
13. APPEND it to the permanent file that was saved in No. 8 above.
14. Log-in under initial User number.
15. Retrieve the file just appended.
16. List it.

EXERCISE 1

This is an exercise in using the permanent file commands. The following areas are emphasized:

-using CATLIST and STATUS to follow what is happening with your files.

-how working files are handled and how they are easily changed and "lost."

-the use of PACK and SORT features.

1. Log on the time sharing terminal and get an abbreviated list of the permanent files saved under your user number using the CATLIST or CAT or CATLIST, LO=Ø or CAT, L=Ø command (all these commands produce the same results).

2. Check to see if the source file that you created in Project 1, Lesson 3, still exists as a permanent file by using the CATLIST,FN=filename command (filename=the name that you gave your file).

3. List and interpret all information on the creation of this file by using the CATLIST,LO=F, FN=filename command.

4. Find out which other users have been using your file by using the command CAT,FN=filename,LO=FP.

5. a. Retrieve several working copies of the file from Project 1, Lesson 3. Each copy should have a different local file name, so use the GET command several times, i.e.:

```
GET,A    =    filename
GET,B    =    filename
GET,C    =    filename
          .
          .
          .
```

   b. Use the ENQUIRE, F command to find out what working files are assigned to your terminal and what mode you are operating in.

   c. Retrieve a copy of your file from Project 1, Lesson 3 as the PRIMARY working file by using the OLD,filename command.

   d. Now, check your status and files again to see what's happened to your working files. What did happen?

6. a. Repeat steps a and b from 5 above.

   b. Repeat step c from 5, above using the ND parameter.

   c. Now check on your files using ENQUIRE, F.

   d. How is this different from d in part 5 above?

7. a. Add the contents of one of the working files to the permanent file from Project 1, Lesson 3 (e.g., APPEND, fn,A).

7. b. List the current primary file.

   c. Retrieve a copy of the permanent file just appended (OLD,fn) using the ND parameter.

   d. List the current primary file including End-of-record marks.

   e. Issue the PACK command to remove embedded EOR's.

   f. Relist the primary file with record marks using the TDUMP,A command.

   g. Note that the EOR embedded in the file has been removed. Note also that the file has not been automatically sorted.

   h. Use the REPLACE command to replace the new lengthened permanent file created by the APPEND, with one of the working files which is an original copy of the file before the APPEND was made (e.g., REPLACE, C=filename.)

8. a. Retrieve as a primary file the file you just replaced in 7, using the OLD command.

   b. Using a line number not currently in the file and less than the line number of the last line of the file add a new line of information to the file.

   c. Issue the NOSORT command.

   d. List the primary file, including record marks, using the TDUMP,A command.

   e. Issue the SORT command specifying no file name, i.e., type SORT.

   f. Issue the NOSORT command.

   g. Relist the primary file, including record marks, using the TDUMP,A command.

   h. Note that the file is still not sorted - SORT when used alone merely resets the sort flag - sort is not actually performed.

   i. Reissue the SORT command specifying no file name.

   j. Relist the primary file, including record marks, using the TDUMP,A command.

   k. Note that the contents of the files are now sorted.

LESSON 5

FILE COMMANDS

LESSON PREVIEW:

This lesson introduces the NOS file manipulation command.

OBJECTIVES:

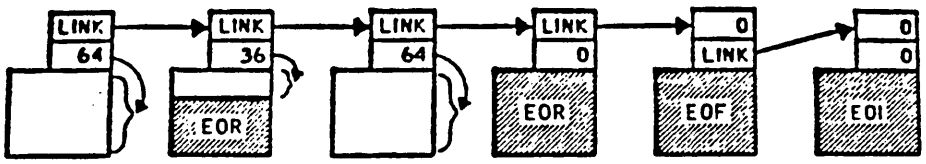
After completing this lesson the student should be able to:

- State the current position on a file before and after specific event
- Use the file manipulation statements, REWIND, SKIPF, SKIPFB, SKIPR, BKSP...
- Use the file copy utilities, COPY, and COPY (BR, BF, CR, CF, EI, SBF and X) to copy portions or all of one file to another

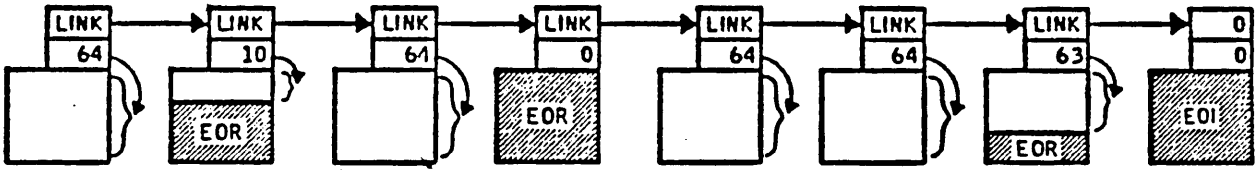
Major File Manipulation Commands

COPY	COPYEI
COPYBR	COPYCR
COPYBF	COPYCF
COPYSBF	COPYX
REWIND	SKIPFB
SKIPEI	BKSP
SKIPR	CLEAR
SKIPF	
RETURN	
UNLOAD	
PACK	
WRITER	
WRITEF	
TDUMP	
VERIFY	

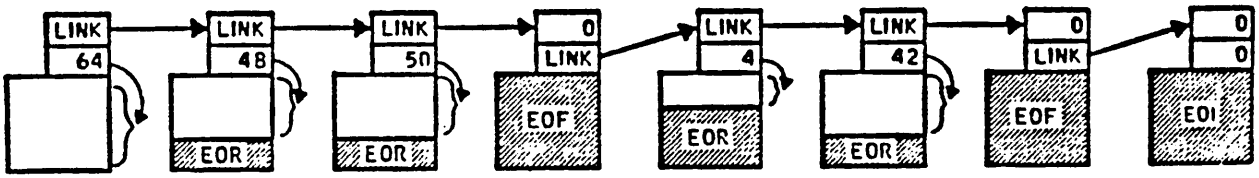
File manipulation commands are described in the NOS Reference Manual.



File 1 consists of 1 file with 2 records.



File 2 contains 3 records.
Notice that it does not contain an EOF.



File 3 is a multi-file.
It contains 2 files, each of which contains 2 records.

Mass Storage File Structure

FILE STRUCTURE INDICATORS

	BEGINNING	LINE IMAGE	SHORT HURDLE	HIGH HURDLE	BRICK WALL
TERMINOLOGY					
OPERATING SYSTEM	BOI	-	EOR	EOF	EOI
CYBER RECORD MGR.	BOI	EOR	EOS	EOP	EOI
PHYSICAL STRUCTURE					
CARDS	FIRST CARD	CARD	7/8/9	6/7/9	6/7/8/9
TAPE (F=SI)	LOAD POINT	-	SHORT PRU + FLAG = 0	SHORT PRU + FLAG = 17_8	TAPE MARK
DISK	FNT POINTER	-	SHORT PRU + LINK TO NEXT PRU	ZERO LENGTH PRU + LINK TO NEXT PRU	ZERO LENGTH PRU; NO LINK
POSITION INDICATORS					
CRM FP FIELD	1	20_8	10_8	40_8	100_8
FTN EOF TEST	EOF=0	EOF=0	INPUT: EOF≠0 other: EOF=0	EOF≠0	EOF≠0
XEDIT	-	-	--EOR--	--EOF--	END OF FILE

EXERCISE 6

PROBLEM

1. Create a local file containing FORTRAN or COBOL source code. Do a CATALOG and TDUMP on this file. (The source code should contain a main program and at least one subroutine.)

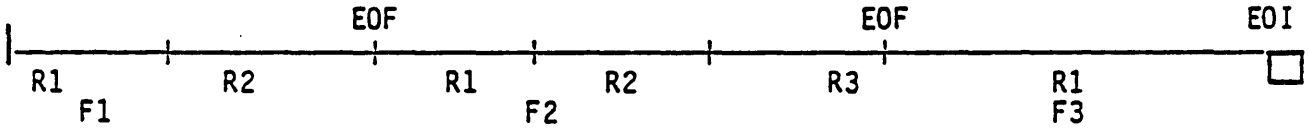
2. Compile the source-code file created in step 1. Do a CATALOG and TDUMP on the relocatable binary file.

3. Use GTR to copy just the subroutine off the binary file. CATALOG the results to check your success.

4. Use COPYX to copy just the main program off the binary file. CATALOG the results to check your success.

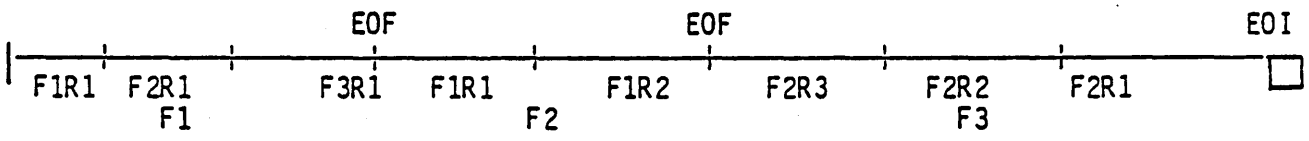
5. Create a file having the indicated structure.

"GIVEN"



6. Use the copy and positioning utilities to create a file with the following structures:

"BUILT"



7. Compare the records in F2 of "GIVEN" file against corresponding records in F3 "BUILT" file.

8. Combine F2 of "GIVEN" file into one logical record.

LESSON 6

XEDIT

LESSON PREVIEW:

This lesson introduces the editing capabilities of the EXEDIT EDITOR.

OBJECTIVES:

After completing this lesson the student should be able to:

- Create a text file
- Manipulate that edit file using the various XEDIT commands

TRAINING AIDS:

Visuals V6-1 through V6-41

PROJECTS:

Exercise 3
Project 2, 3, 4

REFERENCES:

XEDIT Reference Manual, PUB # 60455730

CALL XEDIT

XEDIT - EDIT FILE IS PRIMARY LOCAL. IF NO PRIMARY, CREATION-MODE ENTERED

XEDIT,lfn - EDIT FILE MAY BE NON-PRIMARY

XEDIT,pfn,P - pfn IS MADE LOCAL FROM PERM. FILE. DIRECT-ACCESS ATTACHED IN WRITE-MODE. IF NO PERM. FILE BY THAT NAME, XEDIT ASKS FOR A NAME TO ASSOCIATE WITH EDIT FILE & CREATION MODE IS ENTERED.

XEDIT,lfn,C - IF lfn PREVIOUSLY CONTAINED AT LEAST 1 LINE, THE CONTENTS ARE DESTROYED. THE 1st LINE ENTERED IN CREATION MODE MUST BE A VALID-CREATION-MODE-COMMAND. SUBSEQUENT LINES MAY BE ANY XEDIT COMMAND. CREATION MODE IS EXITED WHEN 1 LINE OF DATA EXITS IN EDIT FILE.

TERMINATE XEDIT

STOP - DOES NOT KEEP EDITED FILE

END,fname,mode

QUIT,fname,mode -

fname - IF NOT SPECIFIED, lfn WHEN XEDIT CALLED IS USED. 'fname' IS NAME OF EDITED VERSION.

mode - L: MAKES EDIT FILE LOCAL. DEFAULT.
S: SAVES AS NEW INDIRECT ACCESS PERM. FILE
R: REPLACES EXISTING INDIRECT ACCESS PERM. FILE WITH EDITED COPY
SL: COMBINES S & L
RL: COMBINES R & L

c: REWRITES EDIT FILE ONTO DIRECT-ACCESS FILE OF SAME NAME. DEFAULT FOR DIRECT ACCESS FILES. IF SAID: DEFINE, ABC, EDIT FILE IS WRITTEN ON DIRECT ACCESS FILE; OTHERWISE ON LOCAL FILE.

XEDIT

```
 XEDIT 3.1.00
 EMPTY FILE/ CREATION MODE ASSUMED.
??
 INPUT
? AAAAAAAAAAAAA
?
 EDIT
?? E
TAPE1   IS A LOCAL FILE

 READY.
```

XEDIT,TEST,P

```
 XEDIT 3.1.00
?? P◆
00100  PROGRAM MINE(OUTPUT)
00110  I=2
00120  J=4
00130  K=I+J
00140  PRINT ◆,I,J,K
00150  END
END OF FILE
?? STOP
 ABORTED

 READY.
```

ENQUIRE,F

```
 LOCAL FILE INFORMATION.

 FILENAME  LENGTH/PRUS  TYPE   STATUS

  INPUT◆                IN.◆   EOR READ
  INPUT                 LO.    I/C READ
  OUTPUT                LO.    I/C READ

 TOTAL = 3

READY.
```

XEDIT,ABABAB,P

```
 XEDIT 3.1.00
 ABABAB NOT FOUND.
NAME EDIT FILE? TOMMY
 EMPTY FILE/ CREATION MODE ASSUMED.
??
 INPUT
? TOMMY IS A GOOD BOY.
?
 EDIT
?? E
TOMMY   IS A LOCAL FILE

 READY.
```

XEDIT,,C

```
 XEDIT 3.1.00
??
 INPUT
? AAAA
?
 EDIT
?? E
TAPE1   IS A LOCAL FILE

 READY.
```

```
GET,TEST


 READY.
LIST,F=TEST

00100  PROGRAM MINE(OUTPUT)
00110  I=2
00120  J=4
00130  K=I+J
00140  PRINT *,I,J,K
00150  END

 READY.
XEDIT,TEST,C

 XEDIT 3.1.00
?? P*
 NULL LINE - IN CREATION MODE
??
 INPUT
? ABABABABABABABABAB
?
 EDIT
?? E
TEST    IS A LOCAL FILE

 READY.
LIST,F=TEST

 79/12/06.  10.02.05.
PROGRAM    TEST

ABABABABABABABABAB

 READY.
```

ON-LINE REF. AIDS

EXPLAIN - REQUESTS EXPLANATION OF ERROR MESSAGE JUST PRINTED.

HELP, COMMAND - REQUESTS INFO. ABOUT A SPECIFIC COMMAND.

INTERRUPTING XEDIT PROCESSING

TO STOP LISTING: (BREAK)
TO RESUME LISTING: (CR)
TO TERMINATE LISTING: (CTRL) (T) AND (CR)
XEDIT WILL RESPOND WITH - ??
TO TERMINATE REQUESTS FOR DATA (ie. ?): (CR)
XEDIT WILL RESPOND WITH - ??

XEDIT CONVENTIONS

?? - XEDIT EXPECTS A COMMAND.
? - XEDIT EXPECTS DATA
"COMMAND SYNTAX": prefix |ln| command n
ln - OPTIONAL # OF LINE AT WHICH COMMAND IS TO BEGIN PROCESSING.
prefix - OPTIONAL CHARACTER WHICH MAY BE USED IN ANY ORDER & COMBINATION
/ : ADVANCES POINTER 1 LINE BEFORE PROCESSING COMMAND
∧ OR ↑ : REPOSITIONS POINTER TO BEGINNING OF FILE BEFORE PROCESSING COMMAND.
+ : INFORMS XEDIT THAT EDITING DATA EXISTS ON SAME LINE AS COMMAND ITSELF WHEN USED IN CONJUNCTION WITH THE DELIMIT COMMAND.

X : TEMPORARILY SUPPRESSES
'VERIFY' OR 'BRIEF' MODE.
OF LINES COMMAND OPERATES UPON.
n : DEFAULT IS 1 - * IS EVERY LINE.

VERIFY - USED TO PRINT LINES CHANGED; USED TO GET OUT OF 'BRIEF'; DEFAULT

BRIEF - WILL NOT PRINT CHANGED LINES; USED TO GET OUT OF 'VERIFY'

THE POINTER

1) WHEN XEDIT IS CALLED, POINTER IS AT BOI.
2) WHEN COMMAND IS PROCESSED, POINTER IS REPOSITIONED AT LAST LINE PROCESSED (USUALLY THE LAST LINE PRINTED IN 'VERIFY' - MODE).
3) WHEN EOI IS REACHED, POINTER MOVES TO BOI.

In # - a) MOVES POINTER TO LINE WITH SPECIFIED NUMBER.
b) LINE IS PRINTED.
c) SEARCH IS CIRCULAR; TOP OF FILE MAY BE PASSED IN ORDER TO POSITION POINTER AT DESIRED LINE.
d) IF NO SUCH LINE EXISTS, POINTER IS MOVED TO LINE WITH NEXT HIGHEST LINE NUMBER.

NEXT n

NEXT - n - ADVANCES OR REVERSES MOVEMENT OF THE POINTER BY n LINES. WILL PRINT LINE POINTER IS AT.

PRINT n - PRINTS n LINES BEGINNING AT POINTER. DEFAULT IS 1. REPOSITIONS POINTER TO n^{th} LINE.

TOP - MOVES POINTER TO TOP OF EDIT FILE. WILL NOT PRINT THE LINE.

BOTTOM - MOVES POINTER TO BOTTOM OF CURRENT LOGICAL RECORD. WILL PRINT THE LINE.

WHERE - GIVE # LINES BETWEEN BOI & CURRENT POSITION OF THE POINTER.

```
XEDIT.HOBO
 XEDIT 3.1.00
?? F◆
AAAA
BBBB
CCCC
DDDD
EEEE
END OF FILE
?? /A2
? XXXXXXXXXX
BBBBXXXXXXXXXX
CCCCXXXXXXXXXX
?? P
CCCCXXXXXXXXXX
?? BRIEF
?? /A
? TTTTTTTTTTTTTTTTTTTT
??  ^F◆
AAAA
BBBBXXXXXXXXXX
CCCCXXXXXXXXXX
DDDDTTTTTTTTTTTTTTTTTTTT
EEEE
END OF FILE
?? V
?? XA
? 888888888888888
?? A2
? WWWWW
AAAA888888888888888WWWWW
BBBBXXXXXXXXXXWWWWW
??  ^F◆
AAAA888888888888888WWWWW
BBBBXXXXXXXXXXWWWWW
CCCCXXXXXXXXXX
DDDDTTTTTTTTTTTTTTTTTTTT
EEEE
END OF FILE
?? STOP
 ABORTED
```

STRING DELIMITERS

1) MUST BE A CHAR. NOT PART OF THE STRING.
2) MUST BE WITHIN THE TIME-SHARING CHARACTER SET.
3) CHARACTER CANNOT BE A SPACE, NUMBER, *, COMMA (OR $ IN IAF).
4) IF AN ALPHABETIC, A SPACE MUST EXIST BETWEEN THE COMMAND & DELIMITER.

NOBELLS - 1) SUPPRESSES BELL RINGING WHEN XEDIT MESSAGES ARE WRITTEN.
2) NO COMMAND TO TURN BELL BACK ON.
3) NO COMMAND TO RING A BELL AT A GIVEN POINT.

LOCATE/STRING/n

LOCATE/STRING1 . . . STRING1A/n - ADVANCES POINTER TO n^{th} LINE CONTAINING THE STRING. ALL n LINES ARE PRINTED.

LOCATE/STRING1 - - - STRING1B/n - ADVANCES POINTER TO n^{th} LINE CONTAINING 'STRING1' BUT NOT 'STRING1B'. ALL n LINES ARE PRINTED. IF 'STRING1' NOT SPECIFIED, XEDIT SEARCHES FOR n LINES NOT CONTAINING 'STRING1B'.

```
XEDIT,SUSAN
 XEDIT 3.1.00
?? P◆
00500 PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
00700 INPUT    X,Y,Z
00800 LET A =  X+Y+Z
00900 PRINT    "X+Y+Z="A
01000 LET B =  X◆Y
01100 PRINT    "X◆Y="B
01200 LET C =  A+B
01300 PRINT    "A+B="C
01400 GO TO 700
01500 END
END OF FILE
?? L/LET/2
00800 LET A =  X+Y+Z
01000 LET B = X◆Y
?? P
01000 LET B =  X◆Y
?? L/PRINT...OR ENT/
END OF FILE
?? P
00500 PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
??  L/PRINT---="B/◆
00500 PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
00900 PRINT    "X+Y+Z="A
01300 PRINT    "A+B="C
END OF FILE
?? STOP
 ABORTED
AEB ,      0.363UNTS.

SBU      0.363 UNTS.
READY.
```

ADD n - ADDS ONE STRING TO ALL n LINES. THE POINTER IS SET TO THE nth LINE.

```
XEDIT,SUSAN
 XEDIT 3.1.00
?? P*
00500 PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
00700 INPUT    X,Y,Z
00800 LET A =  X+Y+Z
00900 PRINT    "X+Y+Z="A
01000 LET B =  X*Y
01100 PRINT    "X*Y="B
01200 LET C =  A+B
01300 PRINT    "A+B="C
01400 GO TO 700
01500 END
END OF FILE
?? 900A3
? /XXXXXX
00900 PRINT    "X+Y+Z="A/XXXXXX
01000 LET B =  X*Y/XXXXXX
01100 PRINT    "X*Y="B/XXXXXX
?? A1
? 888
01100 PRINT    "X*Y="B/XXXXXX888
?? A0
? 999
01100 PRINT    "X*Y="B/XXXXXX888999
?? A
? 777
01100 PRINT    "X*Y="B/XXXXXX888999777
?? P
01100 PRINT    "X*Y="B/XXXXXX888999777
?? W
       7
?? B
01500 END
?? STOP
 ABORTED
AEB ,     0.312UNTS.
READY.
```

REPLACING STRINGS BY CONTEXT

CHANGE/STRING1/STRING2/n
CHANGE/STRING1 . . . STRING1B/STRING2/n - CHANGE EVERY OCCURRENCE OF 'STRING1' TO 'STRING2' WITHIN THE 1st n LINES WHICH CONTAIN 'STRING1'.

CHANGES/STRING1/STRING2/n
CHANGES/STRING1 . . . STRING1B/STRING2/n - CHANGE THE 1st n OCCURRENCES OF 'STRING1' TO 'STRING2', AND PRINT THE LINES WHICH CONTAIN THEM.

```
XEDIT,JOHNNY
 XEDIT 3.1.00
?? P◆
00100 HELLO THERE      HELLO THERE      HELLO THERE
00110 HELLO THERE      HELLO THERE      FIRE
00120 HELLO THERE      HELLO THERE      HELLO THERE
00130 GOODBY NOW       GOODBY NOW       GOODBY NOW
00135 GOODBY NOW       GOODBY NOW       GOODBY NOW
00140       HELLO THERE      HELLO THERE      HELLO THERE
00150       HELLO THERE      HELLO THERE      HELLO THERE
END OF FILE
?? 110C/HELLO THERE/XXX/2
00110 XXX      XXX      FIRE
00120 XXX      XXX      XXX
?? C/FIRE/888/
END OF FILE
?? ^CS/NOW/77777/4
00130 GOODBY 77777       GOODBY 77777       GOODBY 77777
00135 GOODBY 77777       GOODBY NOW       GOODBY NOW
?? ^F◆
00100 HELLO THERE      HELLO THERE      HELLO THERE
00110 XXX      XXX      FIRE
00120 XXX      XXX      XXX
00130 GOODBY 77777       GOODBY 77777       GOODBY 77777
00135 GOODBY 77777       GOODBY NOW       GOODBY NOW
00140       HELLO THERE      HELLO THERE      HELLO THERE
00150       HELLO THERE      HELLO THERE      HELLO THERE
END OF FILE
```

DELETING STRINGS BY CONTEXT, AND INSERTING LINES AT BEGINNING OF A LINE —

```
?? C/BY 777//2
00130 GOOD77       GOOD77       GOOD77
00135 GOOD77       GOODBY NOW       GOODBY NOW
?? P
00135 GOOD77       GOODBY NOW       GOODBY NOW
?? 140CS/THERE//5
00140       HELLO       HELLO       HELLO
00150       HELLO       HELLO       HELLO THERE
?? ^P◆
00100 HELLO THERE      HELLO THERE      HELLO THERE
00110 XXX      XXX      FIRE
00120 XXX      XXX      XXX
00130 GOOD77       GOOD77       GOOD77
00135 GOOD77       GOODBY NOW       GOODBY NOW
00140       HELLO       HELLO       HELLO
00150       HELLO       HELLO       HELLO THERE
END OF FILE
?? C//AAAAA/3
AAAAA00100 HELLO THERE      HELLO THERE      HELLO THERE
AAAAA00110 XXX      XXX      FIRE
AAAAA00120 XXX      XXX      XXX
?? P
AAAAA00120 XXX      XXX      XXX
?? /CS//BBB/2
BBB00130 GOOD77       GOOD77       GOOD77
BBB00135 GOOD77       GOODBY NOW       GOODBY NOW
?? ^P◆
AAAAA00100 HELLO THERE      HELLO THERE      HELLO THERE
AAAAA00110 XXX      XXX      FIRE
AAAAA00120 XXX      XXX      XXX
BBB00130 GOOD77       GOOD77       GOOD77
BBB00135 GOOD77       GOODBY NOW       GOODBY NOW
00140       HELLO       HELLO       HELLO
00150       HELLO       HELLO       HELLO THERE
END OF FILE
?? STOP
 ABORTED
AEB ,       0.411UNTS.
READY.
```

MODIFY - CAN CHANGE ONLY 1 LINE AT A TIME. MODIFY PRINTS THE CURRENT CONTENTS AT THE POINTER AS AN ALIGNMENT GUIDE. MODIFY USES THE FOLLOWING DIRECTIVES:

1) BLANK SPACE - CHARACTER ABOVE REMAINS UNCHANGED.
2) & - DELETES CHAR. ABOVE & REPLACES IT WITH A BLANK.
3) # (BY ITSELF) - DELETES CHARACTER ABOVE & CLOSES UP.
4) ↑ (BY ITSELF) - INSERTS A BLANK IN FRONT OF CHAR. POINTED TO.
5) ↑ STRING # - INSERTS STRING IN FRONT OF CHARACTER ↑ POINTS TO. # IS STRING DELIMITER.
6) OTHER ALPHA/NUM. CHAR. - REPLACES CHARACTER IN THE LINE ABOVE.

```
XEDIT,MOD2
 XEDIT 3.1.00
?? 10M
  00010XTHIS STRING TO BE MORITFD.
?      &     ^IS THE ##      D ## ^IE##
00010 THIS IS THE STRING TO BE MODIFIED.
?? STOP
 ABORTED
AEB ,      0.273UNTS.
READY.
```

QMOD n - CAN CHANGE n LINES AT ONE TIME. PROVIDES A ROW OF COLUMN #'s AS AN ALIGNMENT GUIDE. USES SAME DIRECTIVES AS MODIFY.

```
XEDIT,SUSAN
 XEDIT 3.1.00
?? ^P◆]
00500 PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
00700 INPUT    X,Y,Z
00800 LET A =  X+Y+Z
00900 PRINT    "X+Y+Z="A
01000 LET B =  X◆Y
01100 PRINT    "X◆Y="B
01200 LET C =  A+B
01300 PRINT    "A+B="C
01400 GO TO 700
01500 END
END OF FILE
?? 6000MOD3
  0        1         2         3         4         5         6
  12345678901234567890123456789012345678901234567890123456789012345678901234567890
?   T   & ^XXX::    ::
00T00  RXXXINT    "O ENTER STOP TO TERMINATE THE PROGRAM"
00T00  NXXXPUT    X,,Z
00T00  EXXXT A    X++Z
?? STOP
 ABORTED
AEB ,      0.292UNTS.
READY.
```

NOTE: ATTEMPTING TO MODIFY PART OF A LINE WHICH DOES NOT EXIST (I.E. LINE 700, COL #21 +) WILL GIVE UNRELIABLE RESULTS.

YQMOD n - SAME AS QMOD, EXCEPT DOES NOT PROVIDE AN ALIGNMENT GUIDE.

```
XEDIT,SUSAN
 XEDIT 3.1.00
?? P◆
00500 PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
00700 INPUT    X,Y,Z
00800 LET A =  X+Y+Z
00900 PRINT    "X+Y+Z="A
01000 LET B =  X◆Y
01100 PRINT    "X◆Y="B
01200 LET C =  A+B
01300 PRINT    "A+B="C
01400 GO TO 700
01500 END
END OF FILE
?? YQMOD2
?       ^ADD THIS ◆
00500 ADD THIS PRINT    "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 ADD THIS PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
?? P
00600 ADD THIS PRINT    "OR ENTER STOP TO TERMINATE THE PROGRAM"
?? STOP
 ABORTED
AEB ,      0.292UNTS.
READY.
```

DELETE n - DELETES LINES STARTING AT THE POINTER. POINTER IS POSITIONED AT LINE AFTER LAST DELETED LINE.

DELETE/STRING/n

DELETE/STR1 . . . STR1A/n - DELETE 1st n LINES CONTAINING ENTIRE STRING.

DELETE/STR1 - - - STR2/n - DELETE 1st n LINES CONTAINING 'STR1' BUT NOT 'STR2'.

```
/OLD,SUSAN
/XEDIT
 XEDIT 3.1.00
?? P◆
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT     "OR ENTER CTRL T TO STOP EXECUTION
00700 INPUT     X,Y,Z
00800 LET A =   X+Y+Z
00900 PRINT     "X+Y+Z="A
01000 LET B =   X◆Y
01100 PRINT     "X◆Y="B
01200 LET C =   A+B
01300 PRINT     "A+B="C
01400 GO TO 700
01500 END
END OF FILE
?? 600D
00600 PRINT     "OR ENTER CTRL T TO STOP EXECUTION
?? D/A/2
00800 LET A =   X+Y+Z
00900 PRINT     "X+Y+Z="A
?? D/NUMBERS/
END OF FILE
?? P
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
?? D/IN...=/2
01100 PRINT     "X◆Y="B
01300 PRINT     "A+B="C
?? ^P◆
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00700 INPUT     X,Y,Z
01000 LET B =   X◆Y
01200 LET C =   A+B
01400 GO TO 700
01500 END
END OF FILE
?? D/LET---X/◆
01200 LET C =   A+B
END OF FILE
?? STOP
 ABORTED
```

REPLACE n - 1) DEFAULT FOR n IS 1.

2) ALLOWS REPLACEMENT OF A SPECIFIC # OF LINES WITH THE SAME # OF SUBSTITUTION LINES.

3) TO REPLACE A LINE WITH A BLANK LINE, THE USER MUST ENTER A SPACE BEFORE THE CARRIAGE-RETURN. IF THE SPACE IS NOT ENTERED 1st, XEDIT STOPS PROCESSING REPLACE AND ASKS FOR THE NEXT COMMAND.

XEDIT

```
 XEDIT 3.1.00
?? P◆
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? /R2
? AAAAAAAA
? BBBBBBBB
?? ^P◆
      PROGRAM CHARS(OUTPUT)
AAAAAAAA
BBBBBBBB
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? STOP
 ABORTED

 READY.
```

INSERT n - INSERT n LINES AFTER POINTER
INSERTB n - INSERT n LINES BEFORE POINTER

NOTE: FOR INSERT, POINTER IS MOVED TO THE LAST LINE ENTERED, FOR INSERTS, POINTER IS NOT MOVED

XEDIT,CHARS,P

```
 XEDIT 3.1.00
?? P*
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? N2
   10 PRINT 1000,I-1,I-1
?? IB2
?              1111111
?              2222222
?? N
 1000 FORMAT(1X,02,1X,R1)
?? I3
?                          1111111
?                          2222222
?                          3333333
?? T
?? P*
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
            1111111
            2222222
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
                         1111111
                         2222222
                         3333333
      END
END OF FILE
?? STOP
 ABORTED

 READY.
```

INPUT MODE - ALLOWS USER TO INSERT AN UNSPECIFIED # OF LINES FOLLOWING POINTER.

"METHOD 1" - TO ENTER: (CR) OR INPUT
- TO LEAVE: (CR)

XEDIT,CHARS,P

```
 XEDIT 3.1.00
?? P◆
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? N2
   10 PRINT 1000,I-1,I-1
?? INPUT
 INPUT
?          111111
?          222222
?
?          444444
?
 EDIT
?? ^P◆
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
        111111
        222222

        444444
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
```

```
?? N
      DO 10 I=1,64
??
 INPUT
?    00000
?
 EDIT
?? ^P◆
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   00000
   10 PRINT 1000,I-1,I-1
        111111
        222222

        444444
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? STOP
 ABORTED

 READY.
```

INPUT e ESCAPE CHARACTER

"METHOD 2" - TO ENTER: INPUT e
TO LEAVE: e EDIT

a) THE USER CAN PROCESS XEDIT COMMANDS PREFIXED BY THE ESCAPE CHARACTER; HE IS RESTRICTED, HOWEVER, TO COMMANDS WHICH DO NOT MOVE THE POINTER (SUCH AS REPLACE INSTEAD OF DELETE). ESCAPE CHARACTER MAY BE ANY CHARACTER EXCEPT SPACE OR THE COMMAND DELIMITER SET BY THE DELIMIT COMMAND (TO BE COVERED LATER).

b) IF THE COMMAND ENTERED AFTER THE ESCAPE CHARACTER WOULD MOVE POINTER, "COMMAND NOT VALID" IS PRINTED.

c) IF THE COMMAND SPECIFIES A REPETITION CHARACTER OTHER THAN 'NULL' OR 'ZERO', "ARGUMENT ERROR" IS PRINTED.

```
XEDIT,CHARS,P

 XEDIT 3.1.00
?? P◆
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? INPUT%
 INPUT
?         ONE LINE
?         TWO LINE
?         THREE LINE
? %D
COMMAND NOT VALID
? %R2
ARGUMENT ERROR
? %R1
ARGUMENT ERROR
? %R0
?         REPLACEMENT OF "THREE LINE"; NOTE: R0 = R1.
? %EDIT
 EDIT
?? ^F◆
      PROGRAM CHARS(OUTPUT)
       ONE LINE
       TWO LINE
       REPLACEMENT OF "THREE LINE"; NOTE" R0 = R1.
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
      END
END OF FILE
?? STOP
 ABORTED

 READY.
```

EDITING LINE NUMBERS

a) BEFORE A LINE NUMBERING COMMAND IS PROCESSED, THE POINTER IS SET TO TOP OF FILE.

b) 'BASIC' PROGRAM BRANCH LINE #'s ARE NOT MODIFIED.

ADDLN ln n - ln: # OF 1st LINE. DEFAULT IS 1.
n : INCREMENT. DEFAULT IS 1.

ADDLNS ln n - SAME AS ABOVE, EXCEPT A SPACE IS ALSO ADDED AFTER THE LINE #.

DELETELN - REMOVES EVERY LINE # FROM A FILE. THE # MUST BEGIN IN COLUMN # 1. HAS NO EFFECT ON NON-NUMBERED LINES.

REPLACELN ln n - REPLACES EXISTING LINE #'s.

FBADL n -
a) POINTER IS NOT SET TO TOP BEFORE COMMAND IS PROCESSED.
b) n: # OF 'BAD LINES' TO BE FOUND. A 'BAD LINE' IS ONE WITHOUT A LINE #.
c) SHOULD BE USED IN VERIFY - MODE SO 'BAD LINES' ARE PRINTED.

DBADL n -
a) POINTER IS NOT SET TO TOP BEFORE COMMAND IS PROCESSED.
b) DELETES 'BAD LINES'.

```
XEDIT,TEXT2

 XEDIT 3.1.00
?? P◆
00100 AAA
00110 BBB
      CCC
00120 DDD
EEE
00130 FFF
END OF FILE
?? ALN25 25
END OF FILE
?? P◆
0002500100 AAA
0005000110 BBB
00075      CCC
0010000120 DDD
00125EEE
0015000130 FFF
END OF FILE
?? ALNS 50 50
END OF FILE
?? P◆
00050 0002500100 AAA
00100 0005000110 BBB
00150 00075      CCC
00200 0010000120 DDD
00250 00125EEE
00300 0015000130 FFF
END OF FILE
?? DLN
END OF FILE
?? P◆
 0002500100 AAA
 0005000110 BBB
 00075      CCC
 0010000120 DDD
 00125EEE
 0015000130 FFF
END OF FILE
?? DLN
END OF FILE
?? P◆
 0002500100 AAA
 0005000110 BBB
 00075      CCC
 0010000120 DDD
 00125EEE
 0015000130 FFF
END OF FILE
```

```
?? QMOD◆
  0        1         2         3
  12345678901234567890123456789012

? ::
0002500100 AAA
0005000110 BBB
00075      CCC
0010000120 DDD
00125EEE
0015000130 FFF
END OF FILE
?? DLN
END OF FILE
?? P◆
 AAA
 BBB
      CCC
 DDD
EEE
 FFF
END OF FILE
?? STOP
 ABORTED

 READY.
```

```
XEDIT,TEXT2

 XEDIT 3.1.00
?? P◆
00100 AAA
00110 BBB
      CCC
00120 DDD
EEE
00130 FFF
END OF FILE
?? RLN 2 2
END OF FILE
?? P◆
00002 AAA
00004 BBB
      CCC
00006 DDD
EEE
00008 FFF
END OF FILE
?? LFBL◆
EEE
END OF FILE
?? LDBL◆
EEE
END OF FILE
?? P◆
00002 AAA
00004 BBB
      CCC
00006 DDD
00008 FFF
END OF FILE
?? STOP
 ABORTED

 READY.
```

DLBLANKS n - DELETES LEADING BLANKS STARTING AT POINTER.

TOPNULL - INSERTS A BLANK LINE AT THE BEGINNING OF THE FILE.

```
XEDIT,TEXT3

 XEDIT 3.1.00
?? P◆
00100 AAA
0000000110 BBB
      CCC
00130 DDD
      EEE
00150FFF
END OF FILE
?? 130DLB◆
END OF FILE
?? P◆
00100 AAA
0000000110 BBB
      CCC
00130 DDD
EEE
00150FFF
END OF FILE
?? TN
?? P2

00100 AAA
?? STOP
 ABORTED
```

WEOR - a) WILL PLACE AN 'EOR' IMMEDIATELY IN FRONT OF LINE AT POINTER.

WEOF - a) WILL PLACE AN 'EOF' IMMEDIATELY IN FRONT OF LINE AT POINTER.

b) IF THERE IS NO 'EOR' AT THAT POINT, WF WILL PUT ONE THERE.

DEOR n - a) n: # OF 'EOR' MARKS TO BE DELETED, STARTING AT POINTER.

b) WILL NOT DELETE AN 'EOR' FOUND IMMEDIATELY IN FRONT OF AN 'EOF'.

DEOF n - a) WILL DELETE 'EOF' MARKS, BUT NOT 'EOR' MARKS WHICH ARE IMMEDIATELY PRECEDING.

XEDIT TEXT3

```
 XEDIT 3.1.00
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
00130 DDD
      EEE
00150FFF
END OF FILE
?? 130WF
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
--EOR--
--EOF--
00130 DDD
      EEE
00150FFF
END OF FILE
?? DEOF1
--EOR--
--EOF--
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
--EOR--
00130 DDD
      FFF
00150FFF
END OF FILE
?? DR
--EOR--
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
00130 DDD
      EEE
00150FFF
END OF FILE
```

```
?? 130WR
?? WF
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
--EOR--
--EOF--
00130 DDD
      EEE
00150FFF
END OF FILE
?? DF
--EOR--
--EOF--
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
--EOR--
00130 DDD
      EEE
00150FFF
END OF FILE
?? DR
--EOR--
```

```
?? 130WR
?? WF
?? DR
END OF FILE
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
--EOR--
--EOF--
00130 DDD
      EEE
00150FFF
END OF FILE
?? DF
--EOR--
--EOF--
?? DR
END OF FILE
?? ^P◆
00100 AAA
0000000110 BBB
      CCC
--EOR--
00130 DDD
      EEE
00150FFF
END OF FILE
?? STOP
 ABORTED

 READY.
```

TEOF + - TURNS ON PRINTING OF 'EOF' MESSAGES DEFAULT.

TEOR + - TURNS ON PRINTING OF 'EOF' MESSAGES DEFAULT.

TEOF - - TURNS OFF PRINTING OF 'EOF' MESSAGES.

TEOR - - TURNS OFF PRINTING OF 'EOR' MESSAGES.

TEOF - TOGGLES PRINTING OF 'EOF' MESSAGES.

TEOR - TOGGLES PRINTING OF 'EOR' MESSAGES.

OCTCHANGE OCT1 OCT2 n -

1) 'OCT1' IS OCTAL DISPLAY CODE THAT REPRESENTS AN EXISTING STRING TO BE REPLACED BY 'OCT2', WHICH IS ALSO OCTAL DISPLAY CODE.
2) 'n' IS # OF LINES TO BE PROCESSED, WHICH CONTAINS 'OCT1'. DEFAULT IS 1. IF 'n' IS ZERO, ONLY THE CURRENT LINE IS PROCESSED.

RESTORE - ALLOWS USER TO CANCEL ALL EDITING OPERATIONS PERFORMED SINCE THE FOLLOWING EVENTS TOOK PLACE:

a) 'END OF FILE' MESSAGE
b) TOP COMMAND
c) ↑ COMMAND PREFIX
d) A FIND-LINE-NUMBER (ln) COMMAND IS PROCESSED, CAUSING A CIRCULAR SEARCH.
e) A NEXT - n COMMAND IS PROCESSED. THE POINTER DOES NOT MOVE TO REACH BOI FOR THIS TO BE EFFECTIVE.

```
ASCII
 READY.
XEDIT,TRYIT

 XEDIT 3.1.00
 EMPTY FILE/ CREATION MODE ASSUMED.
??
 INPUT
? TODAY'S PASSWORD IS:
? XXXXXXX$ZZZZZZZ$HHHHHHH$IIIIIII$
? AAAAA
? XXXXXXX$ZZZZZZZ$
? BBBBB
? XXXXXXX$ZZZZZZZ$
? CCCCC
? XXXXXXX$ZZZZZZZ$
?
 EDIT
?? ^P◆
TODAY'S PASSWORD IS:
XXXXXXX$ZZZZZZZ$HHHHHHH$IIIIIII$
AAAAA
XXXXXXX$ZZZZZZZ$
BBBBB
XXXXXXX$ZZZZZZZ$
CCCCC
XXXXXXX$ZZZZZZZ$
END OF FILE
?? 0C 53 7655 0
STRING NOT FOUND
?? 0C 53 7655 1
■■■■■■■
?? ^P◆
TODAY'S PASSWORD IS:
■■■■■■■
AAAAA
XXXXXXX$ZZZZZZZ$
BBBBB
XXXXXXX$ZZZZZZZ$
CCCCC
XXXXXXX$ZZZZZZZ$
END OF FILE
?? STOP
 ABORTED

 READY.
```

```
NEW,TRYIT


 READY.
ASCII
 READY.
XEDIT

 XEDIT 3.1.00
 EMPTY FILE/ CREATION MODE ASSUMED.
?? I2
? TODAY'S PASSWORD IS:
? XXXXXXX$HHHHHHH$ZZZZZZZ$IIIIIII$
?? ^P◆
TODAY'S PASSWORD IS:
XXXXXXX$HHHHHHH$ZZZZZZZ$IIIIIII$
END OF FILE
?? 0C 53 7655 1
■■■■■■■
?? REST
?? ^P◆
TODAY'S PASSWORD IS:
XXXXXXX$HHHHHHH$ZZZZZZZ$IIIIIII$
END OF FILE
?? STOP
 ABORTED

 READY.
```

DEFTAB CHAR - DEFINES THE CHARACTER FOR USE IN ENTERING EDITING DATA WITH INSERT, INSERTB, REPLACE & INPUT - MODE COMMANDS. THERE IS NO DEFAULT 'CHARACTER'.

TABS t_1 t_2 . . t_n - DEFINES TAB-STOP POSITIONS. DEFAULT IS 11, 18 & 30. TABS OR TAB WITHOUT PARAMETERS CLEARS PREVIOUS TAB-STOPS.

LISTAB - LISTS CURRENT TAB CHARACTER & TAB-STOP POSITIONS.

```
XEDIT,NEWFILE

 XEDIT 3.1.00
 EMPTY FILE/ CREATION MODE ASSUMED.
?? DT %
?? TAB 10 15 20
??
 INPUT
?      I. STATISTICS
? %A. PROBABILITY
? %B. EXPECTED VALUE
? %%1. AVERAGE
? %%2. VARIANCE
? %%3. STANDARD DEVIATION
? %C. REGRESSION THEORY
?
 EDIT
?? ^F◆
    I. STATISTICS
          A. PROBABILITY
          B. EXPECTED VALUE
                1. AVERAGE
                2. VARIANCE
                3. STANDARD DEVIATION
          C. REGRESSION THEORY
END OF FILE
?? LT
  % TABS   10   15   20
?? STOP
 ABORTED

 READY.
```

RMARGIN n - a) SETS COLUMN POSITION OF RIGHT MARGIN.

b) 'n' MUST BE $\geq$ 10.

c) LINES EXTENDING BEYOND THIS MARGIN CAN BE LOCATED & TRUNCATED; THESE ARE CALLED "LONG LINES."

d) AFTER COMMAND IS EXECUTED, POINTER RETAINS ORIGINAL POSITION.

FINDLL n - a) 'n' IS # OF LINES LONGER THAN THE CURRENT RM SETTING.

b) POINTER IS POSITIONED AT LAST 'LONG-LINE' FOUND, UNLESS 'EOF' IS REACHED.

TRUNCATE n - a) TRUNCATES A SPECIFIED # OF 'LONG-LINES'.

b) 'n' IS # OF 'LONG-LINES' TO TRUNCATE.

```
XEDIT,TRUNC1

 XEDIT 3.1.00
??^P◆
1234567890123456789012345
AAAAAAA
BBBBBBBBBBBB
CCCCCCCCCCCCCCCCCCCC
DDDD
EEEEEEEEEEEEEE
END OF FILE
?? RM 10
?? FLL2
1234567890123456789012345
BBBBBBBBBBBB
?? TRUNC ◆
BBBBBBBBB
CCCCCCCCCC
DDDD
EEEEEEEEEE
END OF FILE
?? P◆
1234567890123456789012345
AAAAAAA
BBBBBBBBB
CCCCCCCCCC
DDDD
EEEEEEEEEE
END OF FILE
?? STOP
 ABORTED

 READY.
```

DELIMIT CHARACTER

a) THE USER CAN SPECIFY MORE THAN 1 COMMAND IN A SINGLE LINE BY SPECIFYING THE DELIMITER CHARACTER TO BE USED TO SEPARATE THE SEQUENCE OF COMMANDS & EDITING DATA ON THAT SAME LINE.

b) ANY VALID COMMAND CAN BE USED.

c) THE DELIMIT-CHAR. REMAINS IN EFFECT UNTIL ANOTHER DELIMIT IS ISSUED.

d) BOTH COMMANDS & DATA CAN APPEAR WITHIN THE DELIMITED SEQUENCE. TO INCLUDE DATA FOR A COMMAND ON THE SAME LINE, THE USER MUST PREFIX THE COMMAND WITH A '+'; OTHERWISE, DATA IS REQUESTED AS NEEDED VIA A SINGLE '?' PROMPT.

e) TO SPECIFY A COMMA AS A DELIMITER:

DEL,,

f) IF ANY COMMAND IS IN ERROR, ALL COMMANDS UP TO THAT POINT ARE PROCESSED, AND THE REMAINDER ARE NOT.

```
XEDIT,RM01
 XEDIT 3.1.00
?? P2
00220 P.B.COLLINS
00230 710 ELM ST
?? DELIMIT &
?? +XA& ZIP 55722&+I&00240 EXT 726&220P3
00220 P.B.COLLINS
00230 710 ELM ST ZIP 55722
00240 EXT 726
?? STOP
 ABORTED

 READY.
XEDIT,RM01

 XEDIT 3.1.00
?? P2
00220 P.B.COLLINS
00230 710 ELM ST
?? DELIMIT &
?? +A& ZIP 55722&+I&00240 EXT 726&220P3
00230 710 ELM ST ZIP 55722
00220 P.B.COLLINS
00230 710 ELM ST ZIP 55722
00240 EXT 726
?? STOP
 ABORTED

XEDIT,RM01

 XEDIT 3.1.00
?? P2
00220 P.B.COLLINS
00230 710 ELM ST
?? DEL,,
?? XA,+I,00240 EXT 726,220P3
?  ZIP 55722
00220 P.B.COLLINS
00230 710 ELM ST ZIP 55722
00240 EXT 726
?? STOP
 ABORTED

 READY.
```

Z & Y COMMANDS

1) FORMATS: Z $ CMD1 $ CMD2$ . . . $ CMDn
Y $ CMD1 $ CMD2$ . . . $ CMDn

2) IT IS SUGGESTED THAT $ OR ; BE USED AS COMMAND DELIMITERS, AND / AS STRING DELIMITERS. BUT, THE COMMAND DELIMITER CAN BE ANY ALPHA/NUMERIC OR SPECIAL CHARACTER NOT APPEARING WITHIN THE Z OR Y COMMAND, OR THE DELIMITER CHARACTER SPECIFIED IN THE LAST DELIMIT.

3) ANY VALID COMMAND, EXCEPT Z OR Y CAN BE USED.

4) IF A DELIMIT IS INCLUDED IN A Z OR Y, THE DELIMIT 'CHARACTER' TAKES EFFECT ONLY AFTER XEDIT HAS PROCESSED ALL COMPONENT COMMANDS IN THE Z OR Y SEQUENCE.

5) IF ANY COMMAND IS IN ERROR, ALL COMMANDS UP TO THAT POINT ARE PROCESSED, AND THE REMAINDER ARE NOT.

6) THE Z OR Y COMMAND SEQUENCE CAN BE REPEATED BY THE -n COMMAND.

7) FOR Z-COMMAND, XEDIT PRINTS THE COMPONENT COMMANDS OF THE SEQUENCE AS THEY ARE PROCESSED, AND FOR Y-COMMAND, IT DOES NOT PRINT THEM.

XEDIT.AAAGG

```
 XEDIT 3.1.00
?? P◆
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
```

(CONTINUED)

```
?? Z$LOCATE/SMITH/$+ADD$ JR.$DELETE/BEE/
?? LOCATE/SMITH/
00160 Q.E.SMITHM
?? +ADD
00160 Q.E.SMITHM JR.
?? DELETE/BEE/
00190 P.T.BEE
END OF FILE
?? Z;L/SMITH/;+A; JR.;D/BEE/
?? L/SMITH/
00160 Q.E.SMITHM JR.
?? +A
00160 Q.E.SMITHM JR. JR.
?? D/BEE/
END OF FILE
?? Z;L/SMITH/;A;D/BEE/
?? L/SMITH/
00160 Q.E.SMITHM JR. JR.
?? A
? JR.
00160 Q.E.SMITHM JR. JR. JR.
?? D/BEE/
END OF FILE
?? Y;L/SMITH/;A;D/BEE/
00160 Q.E.SMITHM JR. JR. JR.
? JR.
00160 Q.E.SMITHM JR. JR. JR. JR.
END OF FILE
?? STOP
 ABORTED

 READY.
```

```
XEDIT;AAAGG

 XEDIT 3.1.00
?? P◆
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? Z;LOCATE/SMITH/;DELIMIT&;+ADD; JR.;DELETE/BEE/
?? LOCATE/SMITH/
00160 Q.E.SMITHM
?? DELIMIT&
?? +ADD
00160 Q.E.SMITHM JR.
?? DELETE/BEE/
00190 P.T.BEE
END OF FILE
?? LOCATE/SMITH/&+ADD& HELLO&D/BEE/
00160 Q.E.SMITHM JR.
00160 Q.E.SMITHM JR. HELLO
END OF FILE
?? STOP
 ABORTED
```

REPEATING COMMANDS

.n OR -n -

- . PROCESS THE PRECEDING COMMAND (OTHER THAN Z OR Y).
- - PROCESS THE PRECEDING Z OR Y.
- n NUMBER OF LINES THAT THE POINTER SHOULD BE ADVANCED BEFORE PROCESSING THE PREVIOUS COMMAND. DEFAULT IS 1; IF n=ZERO, THE LAST COMMAND IS PROCESSED WITHOUT ADVANCING THE POINTER.

XEDIT,AAAGG

```
 XEDIT 3.1.00
?? P
00130 A.B.MACDONALD
?? .3
00157 A.P.MACDONALD
?? .3
00180 EXT 67
?? ^P◆
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? STOP
 ABORTED

 READY.
```

XEDIT,ADDRESS

```
 XEDIT 3.1.00
?? P◆
NAME:
ADDRESS:
END OF FILE
?? DEL $
?? L/ADDRESS/$+INSERT$ZIP CODE:
ADDRESS:
?? ^PRINT◆
NAME:
ADDRESS:
ZIP CODE:
END OF FILE
?? ADD$.$.$.
?   Q.E. SMITH JR.
NAME:  Q.E. SMITH JR.
?   POB 55
ADDRESS:  POB 55
?   55703
ZIP CODE:  55703
END OF FILE
?? ^F◆
NAME:  Q.E. SMITH JR.
ADDRESS:  POB 55
ZIP CODE:  55703
END OF FILE
?? STOP
 ABORTED

 READY.
```

NOTE: THE PRECEDING Z OR Y DOES NOT HAVE TO BE THE LAST COMMAND PROCESSED.

```
XEDIT,AAAGG,P

 XEDIT 3.1.00
?? P◆
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? Z$LOCATE/SMITH/$+ADD$ JR.$DELETE/BEE/
?? LOCATE/SMITH/
00160 Q.E.SMITHM
?? +ADD
00160 Q.E.SMITHM JR.
?? DELETE/BEE/
00190 P.T.BEE
END OF FILE
?? A
? ◆◆◆
00130 A.B.MACDONALD◆◆◆
?? A
? ##◆
00130 A.B.MACDONALD◆◆◆##◆
??-2
?? LOCATE/SMITH/
00160 Q.E.SMITHM JR.
?? +ADD
00160 Q.E.SMITHM JR. JR.
?? DELETE/BEE/
END OF FILE
?? STOP
 ABORTED

 READY.
```

TRIM+	IGNORES TRAILING BLANKS.
TRIM-	USES TRAILING BLANKS; DEFAULT.
TRIM	TOGGLES THE TRIM SETTING.

1) TRAILING BLANKS ARE THE SPACES AT THE END OF A LINE.
2) LINES THAT ARE ENTIRELY BLANK ARE NOT SEARCHED WHEN XEDIT IGNORES TRAILING BLANKS.
3) YOU CAN USE TRIM WITH:

CHANGE	DELETE
CHANGES	LOCATE
COPY	OCTCHANGE
COPYD	

```
XEDIT,TRIMEX

 XEDIT 3.1.00
?? P◆
EX00100 A.B. KRAMER  HQR24130
EX00110 J.J. JOHNSON SWP19130
EX00130 B.C. MILLER  HQR44130
END OF FILE
?? L/130 /
EX00100 A.B. KRAMER  HQR24130
?? L/130          /
EX00100 A.B. KRAMER  HQR24130
?? TRIM+
?? L/130 /
EX00130 B.C. MILLER  HQR44130
?? ^TRIM
?? L/130 /
EX00100 A.B. KRAMER  HQR24130
?? STOP
 ABORTED

 READY.
```

```
XEDIT,TRIMEX,P

 XEDIT 3.1.00
?? P◆
EX00100 A.B. KRAMER  HQR24130
EX00110 J.J. JOHNSON SWP19130
EX00130 B.C. MILLER  HQR44130
END OF FILE
?? TRIM+
?? CS/130 /140 /
EX00140 B.C. MILLER  HQR44130
?? TRIM-
?? M
  EX00140 B.C. MILLER  HQR44130
?       3
EX00130 B.C. MILLER  HQR44130
?? ^CS/130 /140 /
EX00100 A.B. KRAMER  HQR24140
?? STOP
 ABORTED

 READY.
```

DEFINING A WINDOW

WMARGIN lm rm -

a) lm: COLUMN POSITION OF LEFT MARGIN.

b) rm: COLUMN POSITION OF RIGHT MARGIN.

c) THIS COMMAND RESTRICTS THE SCOPE OF A STRING SEARCH TO A SPECIFIED RANGE OF COLUMNS.

d) LEGAL COMMANDS: (MAY ABBREVIATE)

CHANGEW	COPYDW
CHANGEA	COPYDA
CHANGESW	DELETEW
CHANGESA	DELETEA
COPYW	LOCATEW
COPYA	LOCATEA

d) W: REQUIRES THAT ALL CHAR. OF THE SPECIFIED STRING RESIDE WITHIN THE WINDOW.

f) A: REQUIRES THAT THE 1st CHARACTER OF THE SPECIFIED STRING RESIDE WITHIN THE WINDOW.

```
XEDIT,SUSAN,P

 XEDIT 3.1.00
?? P*
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT     "OR ENTER CTRL T TO STOP EXECUTION
00700 INPUT     X,Y,Z
00800 LET A =   X+Y+Z
00900 PRINT     "X+Y+Z="A
01000 LET B =   X*Y
01100 PRINT     "X*Y="B
01200 LET C =   A+B
01300 PRINT     "A+B="C
01400 GO TO 700
01500 END
END OF FILE
?? WM7 35
?? LW/PRINT...NUMBERS/
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
?? LW/PRINT/3
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
00600 PRINT     "OR ENTER CTRL T TO STOP EXECUTION
00900 PRINT     "X+Y+Z="A
?? ^LW/PR...NUMBERS IN/
END OF FILE
?? LA/PR...NUMBERS IN/
00500 PRINT     "ENTER THREE NUMBERS IN THE FORM X,Y,Z"
?? CW/PR...NUMBERS/XXX/
00500 XXX IN THE FORM X,Y,Z"
?? CSW/Y/********/2
00700 INPUT     X,********,Z
00800 LET A +   X+********+Z
?? STOP
 ABORTED

 READY.
```

COPYING LINES ONTO ANOTHER FILE

COPY fname / string1--string2/n	Copies onto file fname all lines from the current pointer position through the nth line containing string1 not followed by string2 or until the end-of-information mark is read.
COPY fname / --string2/n	Copies onto file fname all lines from the current pointer position through the nth line not containing string2 or until the end-of-information mark is read.
COPY fname n	Copies n lines from the edit file onto file fname
COPY fname / string/n	Copies onto file fname all lines from the current pointer position through the nth line containing the specified string or until the end-of-information mark is read.
COPY fname / string1...string2/n	Copies onto file fname all lines from the current pointer position through the nth line containing string1 followed by string2 or until the end-of-information mark is read.

fname - Name of the file onto which the specified lines are to be copied. At least one blank must be before and after fname.

n - Number of lines to be copied or number of lines containing the specified string that are to be located before the copying is completed. Default is 1.

a. BOTH COPY & COPYD HAVE THE SAME FUNCTION, EXCEPT THAT THE COPIED LINES ARE DELETED FROM THE EDIT FILE WITH COPYD.

b. COPYING BEGINS AT THE CURRENT POINTER POSITION & INCLUDES ALL LINES FROM POINTER THROUGH LINE CONTAINING n^{th} OCCURRENCE OF SPECIFIED STRING.

c. AFTER COPYING, POINTER IS POSITIONED AT LAST LINE COPIED.

d. BEFORE COPYING BEGINS, XEDIT REWINDS THE LOCAL FILE. LATER COPIES ONTO THE SAME FILE ARE ADDED TO THE END OF THAT FILE.

```
XEDIT,AAAGG

 XEDIT 3.1.00
?? F*
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? 0157COPYD NEW01 3
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
?? ^F*
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? STOP
 ABORTED

 READY.
REWIND,NEW01


 READY.
```

```
LIST,F=NEW01



00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT

 READY.
XEDIT,AAAGG

 XEDIT 3.1.00
?? F*
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? COPYD NEW01 /SMITHM/
00160 Q.E.SMITHM
?? ^F*
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? STOP
 ABORTED
```

MERGING FILES INTO THE EDIT FILE

READ fname1 . . . . fnamen - COPIES LOCAL FILES INTO EDIT FILE
READP fname1 . . . . fnamen - COPIES PERMANENT FILES INTO EDIT FILE

a) SPECIFIED FILE IS COPIED ONTO EDIT FILE AFTER CURRENT POINTER POSITION.
b) ONCE COPYING IS COMPLETED, POINTER IS POSITIONED AT LAST LINE COPIED ONTO THE EDIT FILE.
c) XEDIT REWINDS THE SPECIFIED FILES BEFORE AND AFTER THE COPY.
d) IF THE SPECIFIED FILE CANNOT BE READ, AN ERROR MESSAGE IS PRINTED AND THE COMMAND PROCESSING ENDS.

```
XEDIT,AAAGG,P

 XEDIT 3.1.00
?? P◆
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? 157READP CHARS TRIMEX NOTHERE
FILENAM CANNOT BE ACCESSED
?? P
EX00130 B.C. MILLER HQR44130
```

```
?? ^P◆
00130 A.B.MACDONALD
00140 A.D.MACMULLEN
00150 A.E.MILLER
00157 A.P.MACDONALD
      PROGRAM CHARS(OUTPUT)
      DO 10 I=1,64
   10 PRINT 1000,I-1,I-1
 1000 FORMAT(1X,02,1X,R1)
      END
EX00100 A.B. KRAMER  HQR24130
EX00110 J.J. JOHNSON SWP19130
EX00130 B.C. MILLER  HQR44130
00160 Q.E.SMITHM
00170 E.O.SHUNT
00180 EXT 67
00190 P.T.BEE
END OF FILE
?? STOP
 ABORTED

 READY.
```

SAVING AN INTERMEDIATE XEDIT FILE

The FILE command enables the user to save an intermediate version of the edit file. When the FILE command is entered, the following procedure occurs.

1. XEDIT saves a copy of the edit file as specified by the mode parameter.
2. XEDIT prints one or more messages verifying its action.
3. XEDIT requests the next editing command.

The format for the FILE command is:

FILE,fname,mode

fname	Name to be given to the copy of edit file.
mode	Type of NOS file to be created. The possible modes are:

Mode	Description
SAVE	Saves the edit file as a new indirect access permanent file.
REPLACE	Replaces the existing indirect access file.
LOCAL	Makes the edit file a local file; LOCAL mode is illegal for direct access files.
COPY	Copies the edit file to file fname. COPY mode is default for direct access files attached in write mode.
RL	Combines the functions of REPLACE and LOCAL modes.
SL	Combines the functions of SAVE and LOCAL modes.

EXERCISE 3 - XEDIT

1. Use the OLD command to create a working file named KI5Gii in your initials or whatever you want from permanent file KIEX5/UN=.
2. Enter the editor for file KI5Gii.
3. List the entire file.
4. Move the search pointer down two lines.
5. Have XEDIT tell you the number of lines between BOI and current position of pointer.
6. List the line.
7. Move the search pointer to the beginning of the file.
8. Remove the character set "DATA" from every line of the file.
9. Check the position of the search pointer.
10. Move the search pointer to line number 100 of the file.
11. Delete line number 100 of the file.
12. Check for the character Ø in the entire file.
13. Replace every occurrence of the character Ø with the number 0.
14. Check for the character I in the entire file.
15. Replace every occurrence of the character I with the number 1.
16. Move the search pointer to line number 80 of the file.
17. Correct line number 80. The line should read: 80, 100, 16.0, 40.
18. List the entire completed file.
19. Save the completed file.
20. Access the permanent file KIEX51 using the GET command. LIST the file.
21. Access the permanent file KI5Gii.
22. Enter the editor.
23. Merge file KIEX51 into KI5Gii after line 20.
24. List the entire file.

25. Add the following line: 90 100, 17.2, 10. List the file. Note where the lines from file KIEX51 are.

26. Resequence the file using whatever numbering scheme desired.

27. List the entire file.

28. Exit the Editor, make sure your Edit file is permanent.

FILE KIEX5

```
OLD,KIEX5
READY.

LNH
10 DATA,100,10.0,100
20 DATA,100,12.0,100
30 DATA,100,14.0,100
40 DATA,100,16.0,100
50 DATA,100,10.0,80
60 DATA,100,12.0,80
70 DATA,100,14.0,80
80 DATA,100,16.0
100 DATA,100,19.6,46
150 DATA,100,77,-77
READY.
```

FILE KIEX51

```
OLD,KIEX51
READY.

LNH
21 LINE ONE OF KIEX51
31 LINE TWO OF KIEX51
41 LINE THREE OF KIEX51
51 LINE FOUR OF KIEX51
READY.
```

CYBER NOS TTY USAGE
XEDIT

Project No. 2

1. Enter text mode and create a file that is a paragraph of information.
2. Use XEDIT to make changes and corrections to the paragraph.
3. Save it as a permanent file.
4. Create a new file which is a Second Paragraph.
5. Add the Second Paragraph to the original file.
6. Put a title line on the paragraphs, which is separated by at least one blank line.
7. Replace the original file with the changed version.

Project No. 3

1. Create a printable file.
2. Save it as an indirect access permanent file. Enter Batch Mode.
3. Retrieve the file created in No. 1.
4. Get a full listing of permanent file information for that file only.
5. Use COPY to create a new file.
6. Rewind the file.
7. Use COPY to list the file on output.
8. Purge the file.

Project No. 4

Retrieve a working copy of the file TEXT/UN=. Fix up the file--it's full of "typos" and some lines are misplaced.

```
OLD,TEXT
READY.

LNH,R
THE NOS TIME SHARING SYSTEM WAS DEVELOPED BY CONTROL DATA
TO PROVIDE INTERACTIVE FOB PROCESSING CAPABILITIES IN ADDITION
TO THE LOCAL AND REMOTE BATCH PROCESSING CAPABILITIES OF THE
THE
NOS TIME SHARING SUSTEM CONCURRENTLY PROVIDES FOUR

TYPES OF PROCESSING••

     1. LOCAL BATCH
     2. REMOTE BATCH
     3. DEFERRED BATCH
     4. INTERACTIVE TERMINAL
CYBER COMPUTER SUSTEM.  THIS CAPABILITY MAKES THE SPEED AND
COMPUTING POWER OF THE CYBERS AVAILABLE TO MORE USERS.  THE
REMOTE TIME SHARING TERMINAL ALSO TAKES ADVANTAGE OF THE COST/
PERFORMANCE RATIO FO CONTROL DATA'S CYBER COMPUTERS.
NORMAL <---------REMOVE
--EOR--
READY.
```

LESSON 7

FULL SCREEN EDITOR (FSE)

LESSON PREVIEW:

This lesson is intended to give an overview to the use of the Full Screen Editor utility under NOS V2. All the details are not reviewed as this is an introductory course. But the student will be able to use the normal features in both full screen and line editing modes.

OBJECTIVES:

After completing this lesson the student should be able to:

- Use the normal features of the Full Screen Editor in screen mode
- Use the normal features of the Full Screen Editor in line mode

PROJECTS:

None

REFERENCES:

NOS Full Screen Editor User's Guide 60460420

NOS FULL SCREEN EDITOR (FSE)

I. INTRODUCTION

A. Hardware Requirements

The Full Screen Editor (herein referred to as the FSE) is intended only to operate in screen mode on only certain hardware. These include the Control Data Corporation (CDC) 721 Terminal, and in somewhat limited mode the CDC 722, Digital Equipment Corporation (DEC) VT100, Zenith Corporation Z19, and Heath Corporation H19. It will operate in line mode on normal line editing terminals such as the Texas Instrument 745, etc.

Appendix E of the FSE User's Guide gives the proper hardware settings for each of these terminals.

B. FSE Capabilities

FSE enables you to edit sequenced or unsequenced files page-by-page (screen mode) or line-by-line (line mode). Screen mode is available for display terminals and line mode is available on any terminal. The emphasis is on screen mode.

II. SCREEN MODE

A. Getting Started

You must identify to NOS what type of terminal you have. This is done with the SCREEN command:

SCREEN,termtype

termtype identifies your terminal type. The valid entries are:

Entry	Terminal Type
721	CDC Viking 721
722	CDC 722
VT100	DEC VT100
Z19	Zenith Z19 or Heathkit H19

Once entered, the entry remains in effect for your entire terminal session, even though you may change edit modes within FSE. If you forget to enter the SCREEN command before entering FSE, then FSE assumes you have a terminal capable of line edit mode only.

Following the SCREEN command you may enter screen editing mode by the command:

```
FSE,lfn
```

If the lfn does not exist as a local file, one will be created for you.

To move the cursor around the screen, use the arrow keys in the numeric keypad of the CDC 721. These are the "8" for the up arrow, "2" for the down arrow, "6" for the right arrow and "4" for the left arrow. If you hold them down they automatically repeat.

The FSE command has seven possible parameters, all of which are optional. These are:

```
FSE, FN=lfn, OP=access, I=input, L=output,
IP=procedure, WF=workfile.directives
```

This can also be written as positional parameters:

```
FSE,lfn,access,input,output,procedure, workfile,
    directives
```

lfn — specifies the file you want to edit. lfn must be local unless you also specify OP=G. If you default lfn, FSE edits the primary file, if any. If there is no primary file, FSE resumes your last edit session of the same log on. If there was no such prior session then FSE will prompt you for a lfn.

access specifies the character set and/or location of the file to be edited. Valid values are:

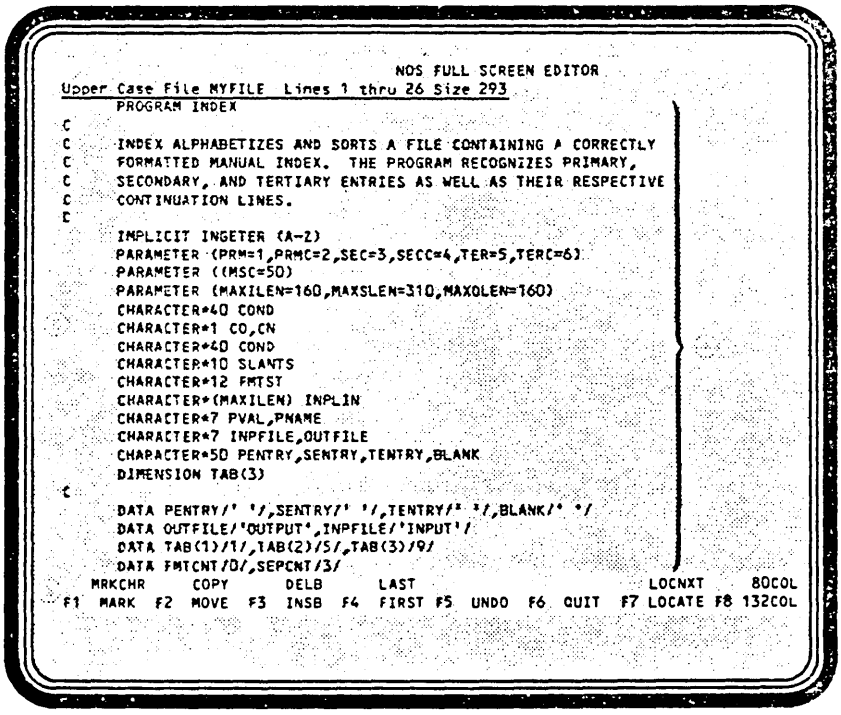
- D 6-bit display code
- N 6-bit display code (default when in NORMAL)
- A NOS 6/12-bit display code (default when in ASCII mode)
- B 7-bit ASCII code right justified in a 12-bit byte
- G Makes lfn local by GETting it if indirect, or ATTACHing it if direct
- R Uses the local copy of the edit file (default). If there have been any changes to the file since you last edited it with FSE, include this READ parameter to access a new copy of the permanent file.

B. Function Keys

You can make most corrections by positioning the cursor and using function keys, or directives can be used.

FUNCTION KEY	DIRECTIVE	DESCRIPTION
INSRT	.INSERT	Inserts a blank or blanks at cursor
INSRT	INSERT	Inserts blank lines
DLETE	.DELETE	Deletes characters
DLETE	DELETE	Deletes lines
FWD	PRINT	Advances the screen display forward
BKW	PRINTPREVIOUS	Previous page is displayed
F2MOVE	MOVE	Moves lines marked by F1MARK to line after cursor position
COPY	COPY	Similar to move without deleting moved lines
F5UNDO	UNDO	Undoes previous operation
F6QUIT	QUIT	Terminate session making all modifications local
	QR	Makes modifications to permanent file
CLEAR		Deletes all characters to end of line
CLEAR		Rewrite screen
F4LAST		Position to last line in the file
F4FIRST		Position to first line in the file
F7LOCATE	LOCATE	Locates occurrence of string
F7LOCNXT		Repeats last search operation
F9MIDDLE	.CENTER	Positions the current line to middle of screen
F10ENDLIN	.END	Moves cursor to the end of the current line
F11SPLIT	.SPLIT	Splits line into two lines
F12JOIN	.JOIN	Joins current line with the next line
F13PARA	.FILL	Reformats lines you marked to conform to margins set with the SET WORD FILL
BACK	BACK	Returns you to a file edited previously

C. SCREEN FORMAT



1. Directive Line. The line on which you enter FSE directives. To position the cursor at this line, press

2. Message Area. The area on the right side of the directive line where FSE displays messages and prompts.

3. File Header. The file name, lines currently displayed, and total number of lines in the file. If the file is uppercase, the prefix `Upper Case` appears on this line. If the file is lowercase, no prefix appears.

4. File Text. The contents of the file.

5. Programmable Function Key Prompts. The labels currently assigned to function keys F1 through F8. These keys are described later in this section.

6. Cursor. Your exact position in the file.

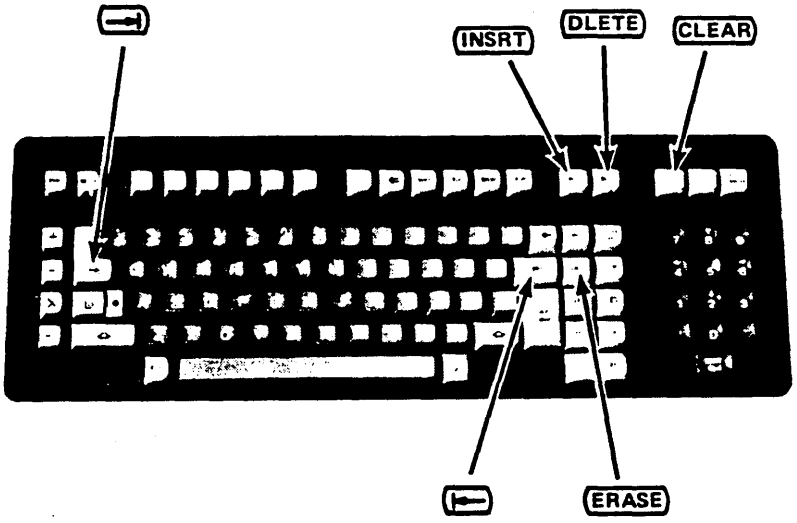
You are now ready to begin editing your file. You can do most of your editing by positioning the cursor to the text to be changed and typing the corrections over the old text. You can perform other editing functions using only the touch of a key. The following sample screen editing session shows you how.

Terminal Function Keys

In addition to the standard ASCII character keys, like (A) and (5), FSE supports three types of function keys: editing, CDC standard, and programmable.

Editing Keys

The editing keys are:



E. Directives

You use directives to tell FSE what to do. In screen mode enter the directives on the directive line, at the top of the screen. In line mode you enter directives after the ?? prompt. In either mode you must press ENTER or NEXT after typing the directive.

Directives are English-like in that they begin with a verb followed by one or more qualifiers called parameters. Some verbs are prefixed by a special character before the verb.

The English-like quality of the directives helps as often you can remember the directive by thinking of it as a sentence. For example:

Sentence	Directive
Locate all occurrences of "abc".	LOCATE ALL "abc"
Move line 12 to just after line 50.	MOVE 12 to 50
Replace the next 3 occurrences "abc" of "abc" with "xyz".	REPLACE NEXT 3 "ABC"XYZ"

You can abbreviate any directive, but not file names, by specifying either the first letter only, or the first three letters only. For example, instead of REPLACE in either uppercase or lowercase, we may have:

```
R
Rep
replace
```

Some of the words can be used in reverse order. For example, above we saw we can LOCATE. In addition to LOCATE ALL WORD /abc/, we could also have:

```
L A W/XVAR/
L/XVAR/A W
L W/XVAR/A
```

These all locate all occurrences of the word XVAR.

Spaces and commas are not required between words; but they are between numbers. This means you can abbreviate even further. For example:

Long version	Compact version
COPY 20 TO 50	C20T50
LOCATE ALL WORD /XVAR/	LAW/XVAR/
MOVE 20 50 TO 100	M20 50T100

1. Directive Parameters

Most directives use a common set of parameters that include direction, strings, and lines. These are:

Parameter	Description
direction	specifies the direction the directive moves through the file as it executes, and the number of times it is to be executed. Valid entries are:

Entry	Description
N num	Begin at the next line and move forward "num" times. Default is 1 time.
P num	Begin at the previous line and move backward "num" times.
R num	Repeat the operating "n" times. The maximum is the number of lines in the file.

(file)	Any local file name. Default is the current file. The parenthesis are required.
line	Specifies the line(s) affected. Default is the current line. Valid entries are:

Entry	Description

	line number	Any line number in the file
	ALL	All lines in the file
	direction	Either of the following with the prior meanings: N num P num
	C	The current line
	F	The first (top) line of the file.
	L	The last line of the file.
	line+num	The line whose number is the sum. For example: 38+45 would be line 83; C+10 would be 10 lines past the current line.
	line-num	Same concept.
	X or Y or Z	Specifies the X, Y, or Z position pointers. Refer to the SET directive.
	line(file)	Specifies the line number shown of the file shown.
range	specifies a range of line to be affected. Default is the current line. Valid entries are:	
	Entry	Description
	line line	A group of lines inclusive of the two numbers.
	M	Lines marked with the F1 MARK key or SET MARK directive.
	S	Lines appearing in the current page on the screen.
string	Specifies a text string. Default is the last string used. Valid entries are:	
	Entry	Meaning

/text/	Specifies the string of text
"text"	Same thing
'text'	Same thing
/text1/... /text2/	Specifies a string of text beginning with characters "text1" and ending with characters "text2", on the same line, and with any intervening characters that lie between

B. PRINT, or P, Directive

The format is:

PRINT (range) (QUIET)

In screen mode it fills up the screen wtih the data following the current pointer position. Thus if we tell it to print line 40 on the screen it will in fact show us lines 40 through 58 (the screen held 19 lines inclusively numbered, in this instance). So in screen mode what we normally want to do is just show one line number; that which starts where we want to view.

Let's look at line editing, and then we will come back to screen mode.

The range we normally want. It specifies what we will print. For example:

P N 3

will print the next three lines following where we are now, but NOT including the current line.

If we want to print three lines including the current we can, by:

```
P  R 3
```

which says repeat printing 3 lines. That prints the current lines and repeats two more times (a total of three) incrementing the line pointer before each repeat.

When finished, the cursor remains at the last line printed.

A range of lines can be printed, for example:

```
P 1Ø 15
```

printed lines 1Ø through 15 inclusive. The pointer remains on line 15.

C. ALTER Directive

This directive enables us to alter, or change, text in the file.

If we enter an A only, FSE responds:

```
<-- ALTER WHAT?
```

at the top upper corner. So first we position the line to where we want to make the change.

Now we type the entry on top to the left. The characters which we enter will replace what is already in the text, character by character. The ampersands (&) is used to change a present character to a blank.

The result of this change can be seen in Figure 7-6.

There are several special characters which we may use, like the ampersand. These are:

Character	Effect
#	Delete a present character, or end an insertion.
&	Replace a character with a space.
!	Delete the present character and the rest of the line. Anything entered after the ! is appended to the end of the shortened line.
caret	The inverted "v", or caret, inserts a character string AFTER the character where the cursor now is.

Let us do something a bit different. We said an exclamation point (!) would truncate a line. So we move the cursor on the same line to the comma after the zero. We enter "A" and press ENTER. The FSE responds with its ALTER message. Since we want to truncate we enter the !.

We can also append text to the end of a line without any prior truncation. This is done using the END parameter, which for the line mode is:

```
A E/end/
```

With the screen mode, let us append to the same line as before, the text "and an end of file". So we type, as shown in Figure 7-11, and are ready to press ENTER.

D. COPY, or C, Directive

COPY copies lines from one place in a file to another place in the same file, or to another file. The full format is:

COPY range (file1) TO line (file2) QUIET

the outer parenthesis indicates the clause is optional.

Let's assume we want to copy lines 84 through 91 on to a new local file we will call XXX. (This often happens, where we want to extract a subroutine or a COBOL PERFORM paragraph and use it in a new program, and we do not want to have to hand code many lines).

E. MOVE, or M, Directive

If we don't want the lines both places, and most of the time we will not, we use a MOVE directive. This works with the same format.

F. GET, or G, Directive

There are two major purposes of this directive. One is to obtain information on the status of our FSE files and FSE function keys. The other is to determine at what column within a line certain items are located. The first is "status"; the second is "align".

G. HELP Directive

A HELP directive is available. We enter H and the directive about which we need help.

H. INSERT, or I, Directive

This allows us to insert new lines into our file, normally following the current line pointer. The full format is:

INSERT line PREVIOUS string BLANK WORD

"P" indicates to insert before the current line pointer. This normally is done only when we are on line 1 and want to insert before it.

Normally, when we INSERT, FSE will respond by "<-- INSERT ?". If we only have a short line to INSERT we may save effort by including the insertion as a string in the directive.

BLANK indicates we want a number of lines, and do not want to resubmit the directive for each line. Once into BLANK we are allowed nine lines before having to "re-ask". Any unused blank lines are deleted by entering DELETE BLANK.

I. LOCATE, or L, Directive

LOCATE locates a specified character string. The full format is:

LOCATE WORD direction string range IN tab
UPPER QUIET

WORD searches for the string as full word, as differentiated from just characters within a word. Any non-alphanumeric characters may constitute the word delimiter, such as parenthesis, hyphens, etc.

Direction is NEXT, PREVIOUS, and REPEAT.

"IN tab" locates a string only within the tab field shown.

UPPER indicates to search as if all the text were capitals. Normally, upper and lower case are treated as different characters.

J. REPLACE, or R, Directive

The REPLACE has the same format as the LOCATE except that now after the location we will replace that string with a new string, and as a result we have two string parameters.

K. SET, or S, Directive

The SET directive has a number of functions, many of which are seldom used by an applications programmer. Several are very useful, however. Among these are:

CHAR	allows the setting of a "software" tab key for when your terminal has no hardware tab key.
TAB	Allows you to reset the tabulation columns. This is nice for building up FORTRAN programs, since the default is columns 7 and 72. However, this is not too conducive for COBOL.
FILENAME	Allows us to rename the current file we are editing.
VIEW	Allows changing the viewing of the screen. It is too detailed to go into here, but if you are attempting to use a CRT terminal not prescribed as a FSE terminal, this would be worth pursuing.
WORD	Used with the advanced features of word processing, allowing centering of headings, etc. Again, persue this at your own leisure; but it can be worthwhile if word processing is your thing.

X, Y, Z — These are three "variables," so to speak, into which you may SET a value. Then the X, Y, or Z may be used in lieu of a line number in any directive where a line number is valid. This allows you the flexibiltiy of a variable, or data-name, within a PROC.

L. UNDO Directive

UNDO "backs out" the changes you have made in a file this session. The backing out is successive in that the first UNDO takes out the last change; the next UNDO the next to last change, etc.

M. Dot Directives

This incorporates a number of "quickie" formats that are very useful. In each case the format is:

```
.param
```

The period and the parameter may be separated by a space if you desire for clarity.

The valid parameters include:

.C — Centers the current line within the preset margins, which by default are 1 and 65.The SET directive can change these defaults.

.D — Delete the current character.

.E — Move the cursor to the end of the current line.

.F — Justifies both left and right margins as in word processing. The default margins of 1 and 65 may be changed by the SET WORD FILL directive. Paragraph indentation is at default column 5.

.I Insert one space at the cursor location.

.I/string Insert the string at the cursor location.

.J Concatenate the current line and the next line.

.P n Move the cursor to column n of the current line.

.S Split the current line at the cursor position.

N. - Directives

The Hyphen, or Dash, is used to call FSE PROCs in from a procedure file. The format is:

-procname (file)

There is always an FSE procedure file with the default name of FSEPROC. If you have not created a local one, or an indirect one, the system has its own direct file. It will take yours first, and its own if it can't find yours.

FSE procedures work just like normal CCL PROCs except there are no parameters, no REVERTs, and the only item in the header is the name of the PROC. Since there is no REVERT you end the PROC either with a QUIT directive or at the EOR. (Remember, CCL PROCs end with an EOR also, which may be the EOF). There are no comment or blank lines.

As a sample, let's assume we want a PROC which will delete all lines in a file that contain a specified string. The following code will do this:

```
DELALL
S A/DELETE WHAT?/
L F/&?/; -DA
-EOR-
DA
D;L;-DA
-EOR-
```

We place this code in our own FSEPROC file. Then we call procedure DELALL by entering:

```
-DELALL
```

The procedure prompts you with DELETE WHAT?. You enter a string and all lines in the current file containing that string will be deleted. When it finds no more matches it will come to the end of your edit file. At that point the PROC will come to its own -EOR- and terminate, and return control back to FSE.

The "&?" is the PROC's own terminology, and these are all listed in the FSE User's Guide.

III. SUMMARY

FSE has a lot of good features, and in its place, it can do a good job. But, like anything new, there are certain problem areas, and consideration areas. Among these are:

1. There is a learning curve. Like all new things it appears very complex at first. But, don't despair.

2. Full Screen Editors are faster than Line Editors. The operator can visually see more of the text at one time, and can envision what to do faster.

3. FSE is available only on NOS V2.2 and later, unless retro-fitted.

4. FSE is designed only for certain hardware.

5. FSE is not intended to replace viable word processors. However, if you have no word processor software/hardware, and only have a minimum of word processing to do, this could be the solution.

6. FSE is new. Like all other new software, expect that it has some glitches. Bear with it and be patient. These will be worked out.

Like all other new software, the best way to get familiar with it, is to start using it.

LESSON 8

BATCH PROCESSING FROM INTERACTIVE TERMINAL

LESSON PREVIEW:

This lesson presents the ROUTE command and associated commands.

OBJECTIVES:

- Use route statement to start a batch job
- Use route to list a file
- Use route to transfer a file to another NOS system
- Use MFQUEUE statement to transfer jobs to other non-NOS systems
- Use MFLINK statement to obtain or transfer files from another system
- Explain concept of LCN

The ROUTE statement queues a * temporary file for disposal to the input queue as a new independent batch job or to the output queue to print, punch, plot, or wait. The routed file can be disposed on the current machine or another NOS machine connected on the LCN (Loosely Coupled Network). The file routing to the output queue may take effect when the statement is processed, a later job step, or at the end of job. To route a file/job to another non-NOS system, the MFLINK/MFQUEUE statement must be used.

Every job or queued output file will have a unique JSN (Job Sequence Number) associated with it. JSN will be used by the system to control jobs.

* Temporary file: The file to be routed must be a local, not a direct access permanent file, and not a primary file. After file disposal, the file is no longer local.

Creating Jobs

The routed file must be in a normal job input format having a valid job card and user card. To route a job to the input queue (create a batch job independent of the terminal job), a disposition code (DC) of either IN, NO, or TO must be used.

At job termination if:

IN: The file output ** would be printed at the default device for the router, unless explicitly routed.

NO: The file output ** would be discarded at job termination unless explicitly routed.

TO: The file output ** would be put in the wait queue for the user owning the job doing the routing unless explicitly routed.

The parameter ST on the route statement or job command for input queue disposition will queue the job to another NOS system in the LCN.

** Output file: The output file is a unique local file per job called OUTPUT. It is a non-ASCII file which at job termination will have a copy of the dayfile as its last record and whose job sequence number (JSN) will be the same as the job's JSN.

Routing Output

Using the route statement, any temporary file may be routed to any type of URD (unit record device) output device by specifying a particular disposition code on a particular device by specifying either ID (identifier id) or FC (format code). ST parameter can specify routing to another NOS system.

Routing may occur immediately or at job termination or later job step if DEF parameter is specified.

The file's OUTPUT, PUNCH, PUNCHB, and P8 are implicitly routed at job termination.

ST=LID specifies the logical identifier of the remote host to which the file is to be routed.

SETJOB

The SETJOB command changes the user job name (UJN), can redirect the disposition of the file output if the job is not interactive or becomes a detached job, and can change the job processing option at job termination or job suspension.

UJN is used with commands such as ENQUIRE, QGET, DROP and others where JSN (Job Sequence Number) could be used to identify jobs or queued files that belong to the user. JSN is a system variable while UJN would be a known value.

The job processing option specifies for detached interactive jobs whether the job is terminated (TJ) or suspended (SU) for a specified system time when an error occurs or the detach job has run out of control cards, or request I/O from terminal.

Example 1:

```
SETJOB, UJN = JOEY
ROUTE, MARY, DC = PR, REP = 3.
   ROUTE COMPLETE.  JSN IS ACPN
```

would route the local file MARY to the printer. Three copies of the file would be printed. Any future reference to drop or status the print file could be done by using JSN = ACPN or UJN = JOEY.

NOTE: The file MARY is no longer local

Example 2:

```
ROUTE, CPUMTR, DC = NO.
   ROUTE COMPLETE.  JSN IS ACPR
```

would create a batch job. At job termination, no output file containing the dayfile would be printed.

Example 3:

```
ROUTE, JOHN, DC = PR, DEF.
   ROUTE COMPLETE.
   .
   .
   .
   .
ROUTE, JOHN.
   ROUTE COMPLETE.  JSN IS ACRB
BYE.

or

ROUTE, JOHN, DC = PR, DEF.
   ROUTE COMPLETE.
   .
   .
   .
   .
   BYE
```

Both sequences of commands will cause the file JOHN to be printed at job termination.

A user may route jobs or files to other NOS systems using the ROUTE statement. To route temporary files or permanent files to other NOS or non-NOS systems, the statements MFQUEUE and MFLINK must be used.

MFLINK statement allows the user to store file on or retrieve permanent files from a remote host. The user may change the size of the file, change the password associated with a file, purge a file or whatever is allowed by the servicer on the remote host. Staging files to another NOS system allows

APPEND	SAVE
CHANGE	ATTACH
GET	DEFINE
PACKNAM	USER
PERMIT	CHARGE
REPLACE	PURGE

For non-NOS file staging, the directives resemble (or, in fact, are) commands for that operating system.

Examples: (NOS to NOS)

A. Assume the user to be at mainframe MFB. To transfer a file from mainframe MFA to MFB and purge it on MFA, the user must enter commands and directives similar to the following, when in batch mode:

```
MFLINK,NEWFILE,ST=MFA.
*ATTACH,OLDFILE/PN=USER.
*PURGE,OLDFILE/PN=USER.
```

B. Assume the user to be on mainframe MFB. To transfer four files from mainframe MFA, on MFB enter commans and directives similar to the following:

```
MFLINK,FILE1,ST=MFA,I=DIRECTS.
MFLINK,FILE2,I=DIRECTS.
MFLINK,FILE3,I=DIRECTS.
MFLINK,FILE4,I=DIRECTS.
```

Where the contents of file DIRECTS are:

```
USER,XYZ,ABC.
CHARGE,CHGN,PROJN.
GET,LFN1.
7/8/9
GET,LFN2.
7/8/9
ATTACH,PFN1/PN=USER.
7/8/9
ATTACH,PFNS.
6/7/8/9
```

MFQUEUE command allows the user to send a temporary file to the input or output queue of the remote host. The MFQUEUE command functions most like the route command. In fact, the ROUTE statement is preferred to CDC mainframes and for most file transfers between NOS and non-CDC mainframes. The MFQUEUE command provides the capability to send a routing directive with the file. Only when the user needs some special capability provided by the routing directive accepted by the remote host does the user require MFQUEUE statement.

Example of routine file to a remote host.

Assume the user to be at mainframe MFA. To route local file RBPRINT from mainframe MFA to a remote batch printer with terminal identifier of AB on NOS mainframe MFB, at mainframe MFA enter statements similar to the following.

```
MFQUEUE(RBPRINT,ST=MFB)
*ROUTE(,DC=PR,TID=AB)
```

The same task could be accomplished with the following ROUTE statement.

```
ROUTE(RBPRINT,DC=PR,TID=AB,ST=MFB)
```

LESSON 9

DETACHING AND RECOVERING JOBS

LESSON PREVIEW:

This lesson will show how to allow the user to run several jobs concurrently.

OBJECTIVES:

After completing this lessson the student should be able to:

- Explain concept of ownership
- Detach jobs
- Obtain detach job status report
- Recover detach job
- Control detach jobs (DROP)
- Use QGET to control and recover output

Queue File Control Statements

Recover

QGET

DROP

Ownership

The owner of a queued file # is allowed to status the file, attach the file, or drop the file.

Ownership of a file is determined by:

1. If an input file is being queued by a subsystem (such as BATCHIO or RBF) the owner of the file is determined by the first USER statement in the job's control statement record.

2. For any other type of queued file, the owner of the file being queued is the owner of the job doing the queuing.

If user AAAA enters a ROUTE or SUBMIT statement which creates a new job having a different user number of BBBB, user AAAA is the owner of the new job and all subsequent output or jobs queued by that job.

NOTE: A queued file is an input file, print file, punch file, executing job or detached job which has a unique JSN (job sequence number).

Example:

A: Assume user CPS4423 is logon and routes file JOHN to the input queue with a different user name. The output file and the file routed by job ABQW both are owned by CPS4423.

```
/ROUTE,JOHN,DC=TO.
ROUTE COMPLETE.  JSN IS ABQW.
/ENQUIRE,JSN.

 JSN SC CS LID STATUS                    JSN SC CS LID STATUS

 ABQO.T.ON.    .EXECUTING                ABQX.B.  .    .WAIT QUE
 ABQW.B.  .    .WAIT QUEUE
```

Where the contents of file JOHN is:

```
ABC.
USER(LAC2172,)
LIMITS,L=JO.
ROUTE,JO,DC=WT,UJN=MARY.
```

A detach job is an interactive terminal job which is no longer connected to a terminal. Detaching a job is available only to interactive users.

A job can be detached at any time during a session using ESC-D (hitting D while escape key is depressed) or Control-D depending on terminal class. Any commands which have been entered as part of a typehead queue immediately before or after the detach will be retained as the job streams for the new job (not the detached job).

A detach job will continue to execute in the system until either:

a) end of control statement reached
b) terminal I/O is attempted
c) time limit or SRU limit is reached

At that point, the system will dispose of the job as specified in the most recent SETJOB statement. The job will either be suspended for the default time period, or be terminated and all files disposed to the specified queue. When a user detaches, the system will relog him in with a new JSN number.

If the system or IAF become unavailable, or user becomes disconected, the job will be automatically detached.

A user may recover (continue) a detached job which he owns by using the recover statement.

EXAMPLE 1

```
    .
    .
    .

/compass,i=a,s=nostext,l=x.
esc-d                                   (to detach)
JOB DETACHED, JSN=ALAB
JSN:ALAC,NAMIAF

RECOVERABLE JOB(S)

JSN       UJN       STATUS      TIMEOUT

ALAB      JOB6      EXECUTING
AANC      JOB3      SUSPENDED   6 MIN.

ENTER GO TO CONTINUE CURRRENT JOB,
      RELIST TO LIST RECOVERABLE JOBS,
      OR DESIRED JSN:  AANC
JSN:  AANC SYSTEM: BATCH  SRU:  10.074
STATUS:
CHARACTER SET:  NORMAL MODES: PROMPT ON
IDLE.  ENTER COMMAND.
/
```

The QGET statement allows the owner of a queued output file (print, punch, plot or wait file) to make the file a local file assigned to his current job.

The local file name of the attached queue file will be JSN value if the parameters UJN or LFN (local file name) are not specified. If UJN is specified and LFN is not, the attach file shall be the UJN value. If the LFN is specified, the local file name will be the LFN value, whether or not JSN or UJN is specified.

```
/QGET, JSN = ACQS, DC = PR, LFN =JOE
 QGET, COMPLETE.
```

ABOVE, lists all the queued file owned by the user and makes the print file ACQS a local file called JOE.

When using UJN, it must be unique.

The DROP statement allows the owner of a queued file (output file or executing job) to remove/terminate the file. A user may not terminate his current job.

Example:

```
DROP, UJN = AAVY, DC = ALL
```

Will drop queued files owned by the user and whose UJN is AAVY in all the queues (output, waiting or executing queues).

The RECOVER statement reconnects a user-owned detached job to the terminal. The current job connected to the terminal is terminated.

The user can recover a specific job with

```
RECOVER, JSN = ALAB.
```

or request recover dialogue with

```
RECOVER.
```

Recover dialogue is initiated during terminal log on if there are recoverable jobs.

Example:

```
ROUTE, GARY, DC = PR.
ROUTE COMPLETE.  JSN IS ACPS.
SETJOB, UJN = MARY, DC = TO.
ROUTE, JOHN, DC = PR, FC = AA, UJN = CPUMT.
ROUTE COMPLETE.  JSN IS ACPT
ROUTE, BILL, DC = PR.
ROUTE COMPLETE.  JSN IS ACPW
ROUTE, CATH, DC = PR
ROUTE COMPLETE.  JSN IS ACPX.
```

Assume about environment for the following examples:

A:

```
QGET, JSN = ACPT, LFN = JOE, DC = PR
QGET, ACPT, PR,, JOE.
QGET, UJN = CPUMT, LFN = JOE, DC = PR.
```

The three commands are equivalent. They would retrieve the file JOHN and make it a local file called JOE.

B:

```
QGET, UJN = MARY, DC = PR.
```

would cause an error because UJN of MARY is not unique.

C:

```
DROP, UJN = MARY, DC = PR.
```

would delete files BILL and CATH.

Seminar Evaluation

Your feedback will be very useful in improving the quality of our seminars!

Seminar Title _____

Dates ___/___/___ to ___/___/___ Location _____

Instructor _____

	Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
CONTENT					
1. The objectives were clearly stated at the beginning of the seminar. Please explain.	SD	D	NS	A	SA
2. All of the seminar objectives were adequately covered. Please explain.	SD	D	NS	A	SA
FACILITIES					
3. The seminar facilities were comfortable. Please explain.	SD	D	NS	A	SA
INSTRUCTOR					
4. The instructor demonstrated a thorough knowledge of the content. Please explain.	SD	D	NS	A	SA
5. The instructor presented the information in a clear and understandable manner. Please explain.	SD	D	NS	A	SA
PARTICIPANT					
6. I had the necessary prior knowledge for this seminar. Please explain.	SD	D	NS	A	SA

7. Indicate your knowledge level of the content prior to this seminar. (Circle one)

No knowledge ← 1 2 3 4 5 6 7 8 9 10 → Knew everythin covered in semina

8. Indicate your knowledge level of the content after this seminar. (Circle one)

No knowledge ← 1 2 3 4 5 6 7 8 9 10 → Knew everythir covered in semin

GENERAL COMMENTS

9. Considering all aspects of the seminar, how would you rate it overall? (Circle one)

Poor ← 1 2 3 4 5 6 7 8 9 10 → Excellent

10. What did you like best about the seminar?

_____

_____

_____

11. What did you like least about the seminar?

_____

_____

_____

12. Please use the space below to make any other general comments about the seminar.

_____

_____

_____

13. Please list colleagues who should receive advance notices of similar seminars.

Name _____ Name _____

Title _____ Title _____

Organization _____ Organization _____

Address _____ Address _____

_____ _____

Phone No. _____ Phone No. _____

14. Would there be enough interested people in your organization to warrant offering the seminar at your site?

Yes ☐ No ☐

If yes, who should be contacted to manage it?

Name _____ Title _____

Organization _____ Phone No. _____

Address _____

Signature _____ Company _____

PARTICIPANT INFORMATION FORM

In order for our seminars/courses to be most effective, they need to take into account the characteristics, needs and objectives of the people who attend them. The information asked for below will assist us in keeping our presentations relevant to the participants and in developing and scheduling new presentations that will meet participant needs. Please complete this form and leave it with the presenter at the next break.

Seminar/Course Title _____ **Date of Presentation** _____

Name _____ **Field or Type of Business** _____

Title _____ **Years of Experience** _____

Business Address _____ **Supervisor's Title** _____

_____ **Last professional degree** _____

_____

List your three primary objectives in attending this seminar.

1. _____
2. _____
3. _____

Will this course/seminar be credited toward certification/training requirements? _____

Rank in order of importance in your choice of this seminar session.

Instructor _____ **Date** _____ **Location** _____ **Employer's Preference** _____

Previous courses/seminars attended relating to this topic.

1. _____
2. _____
3. _____

Topics for additional courses/seminars in which you would be interested.

1. _____
2. _____
3. _____

What trade journals/magazines do you regularly read or subscribe to in order to keep abreast in your profession?

1. _____
2. _____
3. _____

How did you become aware of this course/seminar?

Schedule/Catalogue _____,

Direct Mail Brochure _____,

Recommendations of Supervisor _____,

Recommendation of Colleague _____,

Corporate Training Department _____,

Other _____.

COMMENT SHEET

MANUAL TITLE: NOS INTERACTIVE TERMINAL USAGE

PUBLICATION NO.: FH3000-1 **REVISION:** D

NAME: _____

COMPANY: _____

STREET ADDRESS: _____

CITY: _____ **STATE:** _____ **ZIP CODE:** _____

This form is not intended to be used as an order blank. Control Data Corporation welcomes your evaluation of this manual. Please indicate any errors, suggested additions or deletions, or general comments below (please include page number references).

CUT ALONG LINE

AA3419 REV 4/79 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 8241 MINNEAPOLIS, MINN.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION

National Coordinator
Bloomington Facility (MNA02B)
5001 West 80th Street
Bloomington, Minnesota 55437

CUT ALONG LINE

FOLD

FOLD